Efficient in-network content distribution : wireless resource sharing, network planning, and security

Michele Mangili

NNT : 2015SACLS182

THESE DE DOCTORAT
de
L'UNIVERSITE PARIS-SACLAY,
préparée à l'Université Paris Sud
et du
POLITECNICO DI MILANO,
Dipartimento di elettronica, informazione e bioingegneria

ÉCOLE DOCTORALE N° 580
Sciences et technologies de l'information et de la communication
Doctoral programme in Information Technology - Cycle : XXVIII

Spécialité de doctorat : Réseaux, Information et Communications

Par

M. Michele Mangili

Efficient in-network content distribution: wireless resource sharing, network planning, and security

Thèse présentée et soutenue à Milan, le 15 décembre 2015:

Composition du Jury :

M. Marc Baboulin, Professeur (LRI - Université Paris-Sud), Président
Mme Brunilde Sansò, Professeure (Polytechnique Montréal), Rapporteur
M. Tijani Chahed, Professeur (Telecom SudParis), Rapporteur
M. Albert Banchs, Professeur (Universidad Carlos III Madrid, IMDEA), Examinateur
M. Fabio Martignon, Professeur (LRI - Université Paris-Sud), Directeur de thèse
M. Antonio Capone, Professeur (DEIB - Politecnico di Milano), Directeur de thèse

université PARIS-SACLAY | ÉCOLE DOCTORALE Sciences et technologies de l'information et de la communication (STIC)

Titre : Distribution efficace des contenus dans les réseaux : partage de ressources sans fil, planification et sécurité

Mots clés : réseaux sans fil, distribution de contenu, planification du réseau, optimisation, sécurité

Résumé : Au cours de ces dernières années, la quantité de trafic que les utilisateurs Internet produisent sur une base quotidienne a augmenté de façon exponentielle, principalement en raison du succès des services de streaming vidéo, tels que Netflix et YouTube.
Alors que les réseaux de diffusion de contenu (Content-Delivery Networks, CDN) sont la technique standard utilisée actuellement pour servir les demandes des utilisateurs, la communauté scientifique a formulé des propositions connues sous le nom de Content-Centric Networks (CCN) pour changer la pile de protocoles réseau afin de transformer Internet en une infrastructure de distribution de contenu. Dans ce contexte, cette thèse de doctorat étudie des techniques efficaces pour la distribution de contenu numérique en tenant compte de trois problèmes complémentaires :
1) Nous considérons le scénario d'un réseau hétérogène sans fil, et nous formulons un mécanisme pour motiver les propriétaires des points d'accès à partager leur capacité WiFi et stockage cache inutilisés, en échange d'une contribution économique.
2) Nous étudions le problème centralisé de planification du réseau en présence de caches distribuées et (I) nous analysons la migration optimale du réseau à CCN; (II) nous comparons les bornes de performance d'un réseau CDN avec ceux d'un CCN, et (III) nous considérons un réseau CDN virtualisé et étudions le problème stochastique de planification d'une telle infrastructure.
3) Nous considérons les implications de sécurité sur le contrôle d'accès et la traçabilité, et nous formulons ConfTrack-CCN, une extension de CCN utilisée pour garantir la confidentialité, traçabilité et l'évolution de la politique d'accès, en présence de caches distribuées.

Title : Efficient in-network content distribution: wireless resource sharing, network planning, and security

Keywords : wireless networks, content distribution, network planning, optimization, security

Abstract : In recent years, the amount of traffic requests that Internet users generate on a daily basis has increased exponentially, mostly due to the worldwide success of video streaming services, such as Netflix and YouTube.
While Content-Delivery Networks (CDNs) are the de-facto standard used nowadays to serve the ever increasing users' demands, the scientific community has formulated proposals known under the name of Content-Centric Networks (CCN) to change the network protocol stack in order to turn the network into a content distribution infrastructure. In this context this Ph.D. thesis studies efficient techniques to foster content distribution taking into account three complementary problems:
1) We consider the scenario of a wireless heterogeneous network, and we formulate a novel mechanism to motivate wireless access point owners to lease their unexploited bandwidth and cache storage, in exchange for an economic incentive.
2) We study the centralized network planning problem and (I) we analyze the migration to CCN; (II) we compare the performance bounds for a CDN with those of a CCN, and (III) we take into account a virtualized CDN and study the stochastic planning problem for one such architecture.
3) We investigate the security properties on access control and trackability and formulate ConfTrack-CCN: a CCN extension to enforce confidentiality, trackability and access policy evolution in the presence of distributed caches.

Université Paris-Saclay
Espace Technologique / Immeuble Discovery
Route de l'Orme aux Merisiers RD 128 / 91190 Saint-Aubin, France

Abstract

In recent years, the amount of traffic requests that Internet users generate on a daily basis has increased exponentially, mostly due to the worldwide success of video streaming services, such as Netflix and YouTube. However, the Internet protocol stack was not engineered for this purpose, but it was instead designed as a means of communication between two remote end-hosts, and therefore, this novel usage scenario is opening new challenges to foster efficient in-network content distribution.

While Content-Delivery Networks (CDNs) are the de-facto standard used nowadays to serve the ever increasing users' demands, the scientific community has formulated proposals to change the network protocol stack in order to turn the network into a content distribution infrastructure. These approaches are commonly known under the name of Information-Centric Networks (ICNs) or also Content-Centric Networks (CCNs), and are based on the idea of addressing the contents themselves, rather than writing in the packets the IP address of the host where the information can be found. Among the advantages obtained by using this novel addressing space, the most remarkable is that in-network caching mechanisms can be leveraged in order to boost the content distribution capabilities of the network.

In this context, this Ph.D. thesis provides new contributions to the field, by tackling the general problem of supporting efficient digital content distribution. More in depth, we specifically focus on three complementary problems, namely: 1. wireless resource sharing, 2. network planning and 3. security aspects for content delivery.

This thesis presents our contributions to the field and discusses the key findings we observed while studying this important problem. In particular, our contributions are the following:

1. In Part I, we extensively describe Named-Data Networking (NDN), the instance of ICN we take into account in this study. We introduce the NDN protocol stack, and we provide references to the state of the art concerning the security, routing and mobility properties of this paradigm.
2. In Part II, we take into account the wireless scenario, and we formulate a novel

auction mechanism that is used to motivate wireless access point owners to lease their unused bandwidth and storage capacities, in exchange for economic incentives. This proposal can improve the content distribution capabilities of the network, while making the operator save significant amount of costs to run the servers of the distribution infrastructure.

3. In Part III we study the centralized, off-line network planning problem, in the presence of distributed caches and we tackle (a) the optimal network design for the migration to an ICN, under a single-time slot; (b) we compare the performance bounds of a CDN with an ICN with evolving object popularity, and (c) we consider a virtualized CDN and study the stochastic network planning problem of one such infrastructure.
4. Lastly, in Part IV we consider the security implications on access control and content access trackability, and formulate *ConfTrack-CCN*, a cache-aware mechanism to foster the efficient enforcement of these security requirements, in an ICN.

We hope that this Ph.D. thesis can shed new lights on the relevant problem of content distribution, by showing that the overall network performance and costs can dramatically be reduced by wisely engineering Internet to turn the network into an efficient content distribution infrastructure.

Résumé

Au cours de ces dernières années, la quantité de trafic que les utilisateurs Internet produisent sur une base quotidienne a augmenté de façon exponentielle, principalement en raison du succès des services de streaming vidéo, tels que Netflix et YouTube. Cependant, la pile de protocoles Internet n'a pas été conçue à cette fin, mais elle a plutôt été conçue comme un moyen de communication entre deux hôtes distants, et par conséquent, ce nouveau scénario d'utilisation nécessite de nouvelles techniques pour favoriser la distribution efficace de contenu grâce à Internet.

Alors que les réseaux de diffusion de contenu (Content-Delivery Networks, CDN) sont la technique standard utilisée actuellement pour servir les demandes des utilisateurs, la communauté scientifique a formulé des propositions pour changer la pile de protocoles réseau afin de transformer Internet en une infrastructure de distribution de contenu. Ces approches sont communément connues sous le nom d'*Information-Centric Networks* (ICN) ou également réseaux centrés contenus (*Content-Centric Networks, CCN*), et sont basés sur l'idée d'utiliser les noms de contenus eux-mêmes comme adresse, plutôt qu'écrire dans les paquets l'adresse IP de l'hôte où l'information peut être trouvée. Parmi les avantages obtenus par l'utilisation de ce nouvel espace d'adressage, le plus remarquable est que dans ce réseau, des mécanismes de mise en cache peuvent être exploités très facilement, afin de stimuler les capacités de distribution de contenu du réseau.

Dans ce contexte, cette thèse de doctorat étudie des techniques efficaces pour la distribution de contenu numérique en tenant compte de trois problèmes complémentaires :

1. Nous considérons le scénario d'un réseau hétérogène sans fil, et nous formulons un mécanisme pour motiver les propriétaires des points d'accès à partager leur capacité WiFi et stockage cache inutilisés, en échange d'une contribution économique.

2. Nous étudions le problème centralisé de planification du réseau en présence de caches distribuées et (I) nous analysons la migration optimale du réseau à ICN ; (II) nous comparons les bornes de performance d'un réseau CDN avec ceux d'un

ICN, et (III) nous considérons un réseau CDN virtualisé et étudions le problème stochastique de planification d'une telle infrastructure.

3. Nous considérons les implications de sécurité sur le contrôle d'accès et la traçabilité, et nous formulons ConfTrack-CCN, une extension de CCN/NDN utilisée pour garantir la confidentialité, traçabilité et l'évolution de la politique d'accès, en présence de caches distribuées.

I Named-Data Networking (NDN)

Named-Data Networking (NDN) [13] est l'un des projets financés par la fondation nationale des sciences des États-Unis dans le cadre du programme architecture Internet du Futur, et il est basé sur le projet Content-Centric Networking (CCN) [5, 100], présenté pour la première fois par Van Jacobson en 2006 [13]. Named-Data Networking sera utilisé tout au long de cette thèse comme modèle pour un Information-Centric Network (ICN) et, pour cette raison, Sec. 2.1 donne une présentation de cette architecture.

L'objectif de NDN est de changer la sémantique du service réseau : au lieu d'utiliser les adresses source et destination des hôtes, les paquets NDN contiennent le nom du contenu auquel l'utilisateur souhaite accéder [13]. Afin d'atteindre cet objectif, NDN propose un protocole alternatif pour Internet qui assure la connectivité globale, tout en fournissant seulement l'ensemble minimal de fonctionnalités nécessaires à cette fin, de façon similaire à ce que la couche universel de réseau IP fait aujourd'hui.

Il existe deux types de paquets NDN : 1. intérêts *(Interest packets)*, et 2. données *(Data packets)*. Les demandes règlent les interactions entre les nœuds du réseau, donc le modèle de communication est *pull*-driven. Un nœud envoyant des paquets Interests agit comme un consommateur de contenu, tandis qu'un nœud qui peut fournir des paquets Data se comporte comme un producteur. Les routeurs ont une double responsabilité : 1. ils effectuent le transfert de paquets et 2. ils peuvent aussi se comporter comme caches distribuées.

Chaque nœud NDN effectue la transmission de paquets en se basant sur la présence de trois structures de données :

1. Pending Interest Table (PIT) ;
2. Content Store (CS) ;
3. Forwarding Information Base (FIB).

Le PIT contient la liste des Interests préalablement transmis, mais pour lesquels il n'y a pas encore de réponse. Ce tableau stocke les interfaces à partir desquelles un Interest avait été reçu, pour mettre en œuvre le *transfert de chemin inverse* (reverse-path forwarding) : dès qu'un routeur reçoit un paquet Data, il 1. vérifie le PIT, 2. transmet le paquet sur les mêmes interfaces à partir desquelles Interests de cet objet sont arrivé et 3. supprime l'enregistrement correspondant dans le PIT.

Le Content Store (CS) agit comme une mémoire cache persistante pour le nœud. Lorsqu'un Interest arrive, le routeur d'abord interroge le CS ; en cas de succès, le routeur sera capable de servir directement le paquet Data. Sinon, en cas de défaut, le nœud

va vérifier si le PIT contient un enregistrement correspondant au nom du contenu demandé. Si un tel enregistrement existe, le paquet Interest sera supprimé, et l'interface à partir de laquelle l'Interest a été reçu sera ajoutée au PIT. Au contraire, s'il n'y a pas des enregistrements dans le PIT pour ce contenu, le nœud va 1. ajouter un nouveau record pour l'Interest reçus dans le PIT, et 2. il va transmettre l'Interest vers le producteur de données. En particulier, le prochain nœud sera choisi par le *Forwarding Strategy Layer*, selon les données disponibles dans le FIB.

En raison du "*soft state*" persisté dans le PIT, les paquets de données sont transmis sur le chemin inverse par rapport aux Interests, et cette symétrie supprime la nécessité de fournir les données relatives à la position des hôtes. Pour des raisons similaires, les paquets de données ne peuvent pas boucler dans NDN, tandis que les boucles d'Interests sont évitées grâce à l'état du PIT. Plus en détail, chaque intérêt contient un Nonce généré aléatoirement qui, couplé avec le nom, identifie de manière unique le paquet. Étant donné que les nœuds maintiennent chaque fois le nom de l'intérêt et le Nonce dans le PIT, ils vont détecter et éliminer le bouclage des paquets Interest. Avoir à traiter le soft state dans le PIT est l'une des principales différences entre NDN par rapport aux réseaux IP.

II Enchères inversées conjointe de bande passante et mémoire cache dans les réseaux contenus sans fil

Dans le chapitre 3, nous considérons le problème de distribution de contenus dans un réseau hétérogène sans fil basé sur NDN.

Les clients mobiles peuvent télécharger des contenus numériques publiés par un fournisseur, en se connectant à des points d'accès sans fil détenus par des tiers. Dans ce scénario, le goulot d'étranglement est souvent la connexion backhaul Internet du point d'accès. Afin d'utiliser efficacement le canal WiFi, tout en déchargeant le serveur d'origine du fournisseur de contenu, dans le chapitre 3, nous proposons d'appliquer le paradigme NDN aux communications sans fil. Notre contribution permet de réduire considérablement les coûts et la consommation d'énergie payés par le fournisseur pour servir le contenu ; d'ailleurs, cela permet également d'améliorer l'expérience des utilisateurs en accédant à des contenus numériques en mobilité.

En particulier, nous proposons une stratégie pour stimuler les tiers à louer conjointement leur bande passante inutilisée et le stockage disponible sur leurs points d'accès sans fil NDN. Nous formulons ce problème comme une enchère inversée gérée par un fournisseur de contenu qui veut augmenter le nombre d'utilisateurs atteint par son service et réduire l'énergie utilisée par l'infrastructure de distribution.

Tout d'abord, nous montrons que l'allocation optimale (avec couverture partielle) est NP-difficile, et par conséquent nous fournissons des heuristiques qui garantissent les propriétés d'*individual rationality* et *truthfulness*, et nous comparons leur performance numériquement. Nous évaluons les avantages de nos mécanismes proposés en termes de réduction des coûts énergétiques pour le fournisseur de contenu obtenu en déchargeant son infrastructure à travers les caches, ainsi que la réduction du temps de calcul nécessaire pour exécuter les algorithmes d'allocation.

Enfin, nous analysons également une approche totalement distribuée où les clients mobiles choisissent de façon autonome le point d'accès à utiliser, et nous modélisons ce scénario comme un jeu de congestion, montrant qu'il présente des propriétés souhaitées (par exemple, existence et unicité d'un équilibre de Nash).

III Planification optimale des réseaux orientés contenus

Les premiers prototypes des routeurs NDN ont déjà commencé à apparaître, cependant, afin de migrer vers ce nouveau paradigme, les opérateurs réseaux doivent faire des investissements non négligeables en vue de l'achat de ces nouveaux appareils NDN. Par ailleurs, de nombreux travaux de recherche ont montré clairement que l'adoption de NDN peut favoriser une meilleure distribution de contenu : même en déployant des caches relativement petites, un taux de succès remarquable peut facilement être obtenu. D'autre part une analyse plus approfondie est encore nécessaire pour évaluer les avantages économiques réels auxquels l'adoption d'une telle technique peut conduire pour les opérateurs réseaux.

Pour toutes ces raisons, le chapitre 4 traite le problème de planification du réseau pour la migration vers Named-Data Networking. En particulier, le chapitre fournit des indications quantitatives claires sur les avantages économiques que les opérateurs peuvent obtenir en migrant vers NDN. Notre contribution est de répondre à cette question importante, en formulant un problème de planification du réseau centré sur le contenu, ainsi qu'un modèle d'optimisation pour étudier la migration vers NDN, avec un budget limité.

Notre formulation tient compte de routage du trafic (*traffic routing*) et la mise en cache de contenu (*content caching*). Nous démontrons que le problème d'optimisation est NP-difficile, ensuite nous formulons des heuristiques pour résoudre efficacement ce problème. Une vaste campagne de simulation avec des topologies de réseau réelles montre que notre heuristique réduit le temps de calcul tout en trouvant des solutions proches de l'optimalité, et donc permettant d'aider les opérateurs de réseaux à évaluer les effets d'une migration vers NDN.

IV Distribution de contenu sous évolution de la popularité

Afin de faire face à l'énorme quantité de contenus numériques, les réseaux de diffusion de contenu (Content-Delivery Networks, CDNs) sont actuellement utilisés comme une technologie bien établie pour servir les demandes des clients. Les CDNs fonctionnent comme des superpositions sur la pile protocolaire TCP/IP, afin de favoriser la distribution de contenu, même si Internet n'a pas été conçu spécifiquement à cette fin. D'autre part, le nouveau paradigme de Named-Data Networking (NDN) vise à combler l'écart de ce désalignement en changeant les protocoles de la couche réseau, pour résoudre le problème de distribution de contenu à sa racine.

Dans le chapitre 5, nous utilisons des modèles d'optimisation pour analyser les gains de performance que CDN et NDN peuvent atteindre, en réduisant le montant total du trafic échangé à travers le réseau. Plutôt que considérer la planification à long

terme, comme dans le chapitre 4, nous étudions ce problème en adoptant un modèle de la popularité de contenu variant dans le temps, qui prend en compte le comportement dynamique des utilisateurs.

Nous découvrons que, dans la plupart des cas, CDN conduit à de meilleures performances, en réduisant le trafic que le réseau devrait offrir, alors que NDN devrait plutôt être préférée dans les scénarios où CDN ne peut pas réagir rapidement à l'évolution de la popularité. En plus de ça, nous montrons que des avantages très limités peuvent être obtenus en modifiant les algorithmes de remplacement de cache.

V Planification stochastique pour les réseaux virtuels de distribution de contenu

Comme indiqué dans la Sec. 4.1, les premiers prototypes pour les routeurs supportant *Named-Data Networking* ont déjà commencé à apparaître. Cependant, malgré le fait que NDN peut stimuler la capacité de distribution de contenu du réseau, son adoption dans le monde entier exige de changer la pile protocolaire utilisée par les deux *end-points* ainsi que des nœuds de réseau intermédiaires et, pour cette raison, il est peu probable que cela se produira dans le futur proche. Cependant, le réseau nécessite encore de nouvelles techniques pour distribuer efficacement des contenus numériques de manière efficace. En particulier, dans le chapitre 6, nous présentons une proposition d'évolution pour les CDN, compatible avec la pile protocolaire actuelle et qui ne nécessite pas de changement dans les end-points.

Les réseaux de diffusion de contenu (Content-Delivery Networks, CDN) ont été identifiés comme l'un des cas d'utilisation pertinents où le paradigme émergent des virtualisation des fonctions réseau (Network Functions Virtualization, NFV) sera vraisemblablement bénéfique. En fait, la virtualisation favorise la flexibilité, puisque l'allocation des ressources de nœuds virtuels CDN permet de faire face à des changements brusques du trafic. Cependant, il y a des cas où doivent toujours être préférés des appliances physiques, par conséquent, nous envisageons une architecture mixte entre ces deux solutions (*physiques* et *virtuelles*), capables d'exploiter les avantages de chacun d'eux.

Motivé par ces raisons, dans le chapitre 6, nous formulons un modèle stochastique de planification qui peut être utilisé par les opérateurs CDN pour trouver la décision optimale à long terme pour la planification du réseau. Le modèle peut être utilisé pour savoir s'il est préférable de déployer des appliances physiques CDN dans le réseau et/ou de louer des nœuds CDN virtuels dans les centres de traitement de données. Les principales conclusions montrent que, pour une large gamme d'options de tarification et profils de trafic, NFV peut réduire considérablement les coûts des opérateurs pour fournir le service de distribution de contenu.

VI Confidentialité, traçabilité et évolution de la politique d'accès

La spécification NDN prévoit que les producteurs de contenu signent cryptographiquement chaque paquet Data afin de garantir *l'intégrité*, *la provenance* et *la pertinence* du contenu (Sec. 2.2). Il suggère également de faire respecter la *confidentialité* en chiffrant le contenu lui-même, et en fournissant les clés de déchiffrement correspondantes seulement aux consommateurs légitimes.

Cependant, la présence de caches pose de nouveaux problèmes liés à l'évolution de la politique d'accès, puisque les producteurs peuvent même perdre le contrôle sur les données qu'ils fournissent. En fait, dans les réseaux orienté contenus, les producteurs doivent faire face à de nouvelles difficultés : 1. il devient très complexe de révoquer les privilèges d'accès une fois que les clés de déchiffrement ont été fournies à des utilisateurs légitimes et le contenu a été mis en cache ; 2. chaque fois que les caches répondent directement aux demandes des consommateurs, les producteurs ne reçoivent aucune indication sur l'accès et, par conséquent, ils peuvent difficilement garder une trace du contenu accède. Ces deux problèmes se posent pour le fait que les caches intermédiaires peuvent persister (et servir) le contenu, même s'il est chiffré avec une clé qui a ensuite été révoquée.

Pour toutes ces raisons, le chapitre 7 traite trois propriétés de sécurité importantes : *confidentialité*, *traçabilité* et *évolution de la politique d'accès*.

Notre contribution est de proposer ConfTrack-CCN, une extension de CCN/NDN utilisée pour fournir un mécanisme cryptographique efficace conçu pour garantir la confidentialité, traçabilité et l'évolution de la politique d'accès, en présence de caches distribuées. ConfTrack-CCN assure conjointement toutes ces trois propriétés en protégeant les données avec deux couches de chiffrement, dont la dernière évolue en fonction des changements de privilèges d'accès. Une interaction forcée consommateur-producteur est utilisée pour télécharger les clés de chiffrement, tandis que les consommateurs eux-mêmes authentifient et fournissent des informations d'accès aux producteurs.

Nous évaluons ConfTrack-CCN en effectuant une campagne de simulation approfondie avec des topologies de réseau réelles. Les résultats montrent clairement que, en moyenne, ConfTrack-CCN assure un taux de succès 20% plus élevé que les autres systèmes de sécurité, tout en introduisant une charge de calcul négligeable.

VII Résumé des contributions

Dans cette thèse, nous avons étudié le problème général de distribution de contenu, et nous avons utilisé le nouveau paradigme de réseaux orientés contenus, en se concentrant sur l'architecture connue comme Named-Data Networking (NDN), pour fournir une réponse efficace à une telle nécessité. En particulier, après avoir fourni dans la partie I une description détaillée sur les principaux aspects concernant ICN, nos contributions étaient sur trois grands thèmes : 1. réseau sans fil orientés contenus (Partie II) ; 2. planification du réseau pour la distribution de contenu (Partie III) et 3. la sécurité dans NDN

(Partie IV). Nos contributions adressent spécifiquement l'efficacité de distribution en réseau de contenus : avec notre travail nous voulions jeter quelques lumières sur l'évaluation des avantages que l'architecture ICN peut avoir, surtout lorsqu'on la compare aux méthodes standard de distribution de contenu.

Notre proposition pour utiliser NDN dans les réseaux sans fil (Chapitre 3) a été spécifiquement conçue pour réutiliser la bande passante et le stockage caches inutilisé à des points d'accès WiFi résidentiels. Dans ce contexte, nous avons proposé un mécanisme d'enchères pour rémunérer les propriétaires de point d'accès sans fil prêts à louer leurs ressources inexploitées. Nous avons imaginé le scénario où des incitations économiques sont payées par le fournisseur de contenu afin d'étendre son réseau de distribution, offrant un accès à large bande aux clients mobiles.

Lors de la formulation de nouveaux modèles de planification pour la distribution de contenu, nous avons pris en compte les différents problèmes de 1. placement des objets ; 2. routage des demandes et 3. placement de serveurs de réplique.

Dans ce contexte, nous avons donné des contributions différentes, en commençant au chapitre 4 à décrire l'étape de migration vers ICN. Plus en détail, nous avons formulé un modèle d'optimisation à single tranche de temps pour découvrir les avantages économiques globaux que l'opérateur doit attendre en raison d'une migration vers un tel paradigme réseau. Dans le chapitre 5, nous avons étendu le modèle présenté dans le chapitre 4, en tenant compte de l'évolution de popularité du contenu. Par ailleurs, nous avons concentré notre analyse sur une comparaison des performances entre réseau de diffusion de contenu (CDN), par rapport à celle d'un réseau orientés contenus. Enfin, au chapitre 6 nous avons étudié une infrastructure virtualisée de distribution de contenu basée sur le nouveau paradigme de virtualisation des fonctions réseau.

Les propriétés de sécurité du CCN ont été prises en compte dans le chapitre 7 où nous avons étudié le contrôle d'accès et la traçabilité, en présence de caches distribuées. Afin de répondre à ces exigences, nous avons formulé un mécanisme de sécurité nommé ConfTrack-CCN, qui implémente ces exigences de sécurité en utilisant du chiffrement symétrique et des fonctions d'hachage standard.

Acknowledgments

First and foremost I would like to sincerely thank my advisors, Fabio and Antonio, for their guidance and all the invaluable suggestions they provided me during these three years. It was a great pleasure and honor to work with both of you, and without your supervision, I would hardly have been able to finish this thesis.

I am also grateful to Stefano Paraboschi, for suggesting me the opportunity to pursue this PhD.

I sincerely thank both Stefano Paris and Federico Malucelli, whose contributions were fundamental to complete chapters 3 and 4 of this manuscript. I would also like to thank Andrea Araldo and Dario Rossi for their fruitful cooperation and discussion, on the topic of cost savings.

A special acknowledgment goes to Brunilde Sansò and Tijani Chahed, who carefully reviewed the thesis and provided extremely valuable comments and suggestions that improved the quality of the manuscript. I would also like to thank Marc Baboulin and Albert Banchs who accepted to be members of the jury, it is a great honor and pleasure for me.

To all those who shared a part of their time and knowledge in the accomplishment of this thesis: I am very grateful for your help. In particular, my gratitude extends to all the people I met at LRI, DEIB, Princeton as well as the good old friends: Andrea, Andrew, Dario, Daniele, Davide, Stefano, Simone, Paolo, Luca, Carmelo, Jocelyne, Ilario, Giulia, Giray, Srinivas, Sarthak. Not only it was awesome to spend a great time with you all, but our discussions made me learn a lot and have contributed to shape this manuscript the way it is.

Finally, a warm and most sincere thank goes to all my relatives, but especially my family, Nella, Maurizio and Alessia: thank you for being always there, when I needed you the most.

Contents

CHAPTER 1

Introduction

It is undeniable that Internet, as we know it today, has undergone very deep changes through recent years: originally composed of only few nodes [119], its worldwide diffusion has been so remarkable that it quickly reached our households, and then became an indispensable means of communication available everywhere, at any time, in the palm of our hands [140]. Applications that only a few years ago seemed to be pure fiction are now taken for granted. As an example, consider that Netflix has been streaming videos for only 7 years, whereas YouTube celebrated in 2015 its tenth anniversary [12, 43, 68]. So far, there are no signs that this growing trend will break; on the contrary, the rise and success of connected home-devices, and the advent of the Internet of Things, will likely make this trend be even more remarkable in the future [6, 160].

As a result of these momentous changes, there seem to exist a misalignment between the way people exploit the network today and the original design goals that drove the development of the Internet protocol stack.

Conceived in the late '60s, Internet was designed to solve the problem of resource sharing [119]: in its dawn, the objective of the network was to let the scientific community remotely exploit and share the computational power which was very expensive at that time, and therefore the protocol stack was specifically engineered for a *communication infrastructure* [41]. For this reason, IP packets contain the address of the remote location *where* the information can be found, although, on the other hand, nowadays Internet users care mostly about *what* content is carried by the packets themselves [100].

In terms of bandwidth requirements, apps download, music streaming and photo sharing through social networks have all certainly had a remarkable impact [50], however, with no doubts the game changer has been video distribution, which nowadays accounts for more than 50% of the overall downlink traffic, in fixed access for North

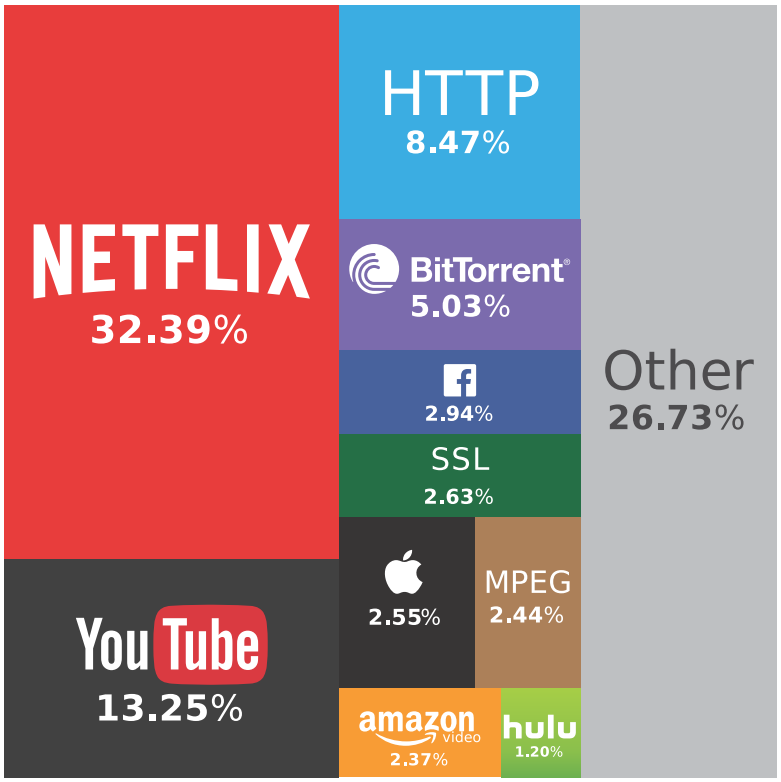


Figure 1.1: Global Internet Phenomena Report (2nd Half 2014). *Per-application downlink traffic share, North America, Fixed Access [160].*

America as in Fig. 1.1 [160].

To support the efficient and effective network distributions of digital contents, specialized technologies such as Content-Delivery Networks (CDNs) were specifically designed to attain this precise objective [141, 191]. Rather than making the origin server fulfill all the incoming requests, in a CDN copies of popular contents are replicated on surrogate servers deployed closer to the locations where users are actually demanding the data [81]. Whenever a host requests one of these popular contents, the CDN infrastructure can transparently make its request be directly processed by a nearby CDN surrogate server, leading to two main advantages: 1. the origin server is offloaded and 2. the end-to-end latency is severely reduced, thus boosting the QoE especially for live streaming [41]. CDN is a technology that was developed as an overlay on top of the TCP/IP protocol stack, in a way such that no dedicated support should be provided by the end-points, and yet, the lack of support from the the network-layer protocol only confirms that these designs were an afterthought to make Internet content distribution scale [191].

At the network layer, from the early ages of Internet until nowadays, the IP protocol continues to survive, showing that the original Internet design goals were so general and universal, that not only there was not the need to evolve the protocol, but IP itself fostered the unprecedented development we discussed so far.

Not only IPv4 is a success story, but the slow adoption of his successor IPv6 [56]

clearly shows the reluctance of changing this basic network primitive. Despite the fact that there are many reasons for the success of IP, the most remarkable key design choice was the *hourglass model* [13]. IP was conceived in a way such that it could be applied in "*the middle of*" the protocol stack: it is independent from link layer protocols, it offers a datagram packet forwarding functionalities, and can be used regardless of the transport protocol up to the application layer level [100]. The hourglass model not only has made possible to build on top of IP the minimal set of functionalities needed for global connectivity, but it has also enabled the explosive Internet's growth letting upper and lower protocol layers evolve independently. Yet, IP packet headers contain host addresses and have nothing to do with contents.

However, in recent years, the research community has started to question whether the misalignment between the protocol stack and users' needs should be resolved, and, to do so, it began to explore alternative research directions in which a better protocol stack is taken into account for the sake of boosting the efficiency of in-network content distribution [18, 41, 191].

This research effort is often motivated by the practical problem that the services provided by a content delivery network are usually very expensive [189], whereas remarkable savings would be obtained if we only could design a better network layer protocol. These type of solutions are known under the name of Information Centric Networks (ICNs), but we will often call them also Content-Centric Networks (CCNs), to emphasize the fact that they are specifically tailored to support the content distribution problem at its roots [59, 112].

Many different designs for Information-Centric Networks have been formulated in the literature: DONA [112], PSIRP/PURSUIT [73], NetInf [59], MobilityFirst [163], Named-Data Networking NDN/CCN [100] just to name a few. Despite the fact that these designs are very different from one another, they all share, as a common feature, the fact that they change the addressing space: the packets themselves contain the name of the content, rather than the host address of the machine providing the corresponding data [41]. As a result of this choice, it becomes much easier to implement a distributed caching mechanism and, in some designs like NDN/CCN, potentially any network node (including intermediate routers) can behave as a cache, thus the content distribution capabilities of the network will be boosted [191].

Information-Centric Networks promise to introduce many benefits such as 1. better network performance though pervasive in-network caching, as well as native support for multicast communications, 2. improved user-experience in mobility, and 3. a more robust security model.

However, new research challenges arise in this context [18, 41, 84, 191]. Without pretending to be exhaustive, among these novel issues we mention the fact that network hardware appliances must be updated to provide adequate support to wire-speed packet forwarding, using the new ICN paradigm [179]. Moreover, routing protocols must be changed in order to support the new content-based addressing space [192]. Although better client's mobility is seamlessly handled in ICN, producers' mobility demands for specific techniques to be managed [175]. Furthermore, from the security perspective, new issues may arise, in fact packet names may disclose important information concerning the content of the communication, thus violating privacy requirements [71].

1.1 Thesis outline and contributions

The ambition of this Ph.D. thesis is to hopefully shed new light on the important problem of network content distribution. More in depth, we provide the following main contributions:

1. In Part I, chapter 2 provides an extensive background on Named-Data Networking (NDN), the instance of Information-Centric Network that we consider. Chapter 2 contains a description of the NDN architecture and provides a detailed illustration of the state of the art for relevant networking problems of this infrastructure.
2. In Part II, chapter 3 considers the content distribution problem in a mobile scenario. We take into account the presence of distributed caches in residential WiFi access points and we formulate a mechanism to provide economic incentives to users' leasing their unexploited bandwidth and cache resources to a content producer. In this context we study game theoretic problems related to one such topic, focusing on the properties of the allocation mechanism we designed, as well as the non-cooperative behavior of mobile clients while connecting to the available access points.
3. In Part III, we study different aspects of the general network planning problem for content distribution, and we formulate novel optimization models for the joint object placement and routing problem. In particular, chapter 4 deals with the long-term planning for the migration to ICN, based on an estimate of the average future traffic requests (single time-slot scenario). Chapter 5 considers a multi-time slots scenario under evolving object popularity, and provides novel results related to the performance comparison of ICN with respect to CDN. Lastly, in chapter 6 we consider a virtualized Content Delivery Network (vCDN) and we tackle the stochastic network planning problem under uncertain traffic demands for one such networking infrastructure.
4. In Part IV, chapter 7 considers the security implications on the content access control and trackability, under access policy evolution, for an Information-Centric Network, and we formulate *ConfTrack-CCN*, an efficient mechanism to enforce these security properties. Finally, concluding remarks are presented in chapter 8.

The common aspect shared by all these contributions is the fact that our solutions are specifically tailored to efficiently solve the problem of in-network content distribution.

1.2 Publications

The contributions discussed in this dissertation have led to the following scientific publications:

Journal Papers

1. M. Mangili, F. Martignon, S. Paraboschi. *A Cache-Aware Mechanism to Enforce Confidentiality, Trackability and Access Policy Evolution in Content-Centric Networks.* Computer Networks, Elsevier, Volume 76, 15 January 2015, Pages 126-145, ISSN 1389-1286.
2. M. Mangili, F. Martignon, A. Capone. *Performance Analysis of Content-Centric and Content-Delivery Networks with Evolving Object Popularity.* Computer Networks, Elsevier, Under Submission. Major revision, June 2015.
3. M. Mangili, F. Martignon, A. Capone. *Optimal Design of Information Centric Networks.* Computer Networks, Elsevier, Under Submission. Minor revision, June 2015.
4. M. Mangili, F. Martignon, S. Paris, A. Capone. *Efficient and Truthful Auction Mechanism for Bandwidth and Cache Leasing in Wireless Information Centric Networks.* IEEE Transactions on Vehicular Technology, Under Submission, May 2015.

Conference Papers

1. M. Mangili, F. Martignon, A. Capone. *Stochastic Planning for Content Delivery: Unveiling the Benefits of Network Functions Virtualization.* The 22nd IEEE International Conference on Network Protocols, ICNP, Raleigh, NC, USA, October 21-24, 2014. Concise Papers Track (acceptance rate: 18.9%, 15 papers out of 79 submissions)
2. M. Mangili, F. Martignon, A. Capone, F. Malucelli. *Content-Aware Planning Models for Information-Centric Networking.* IEEE Global Communications Conference, GLOBECOM 2014, Austin, TX, USA, December 8-12, 2014. **Among the Best 50 papers at Globecom 2014 (2171 submitted papers - 867 accepted)**
3. A. Araldo, M. Mangili, F. Martignon, D. Rossi. *Cost-aware Caching: Optimizing Cache Provisioning and Object Placement in ICN.* IEEE Global Communications Conference, GLOBECOM 2014, Austin, TX, USA, December 8-12, 2014.
4. M. Mangili, F. Martignon, A. Capone. *A Comparative Study of Content-Centric and Content-Distribution Networks: Performance and Bounds.* IEEE Global Communications Conference, GLOBECOM 2013, Atlanta, GA, USA, December 9-13, 2013. **Best Paper Award (2272 submitted papers - 841 accepted)**
5. M. Mangili, F. Martignon, S. Paris, A. Capone. *Efficient Joint Bandwidth and Cache Leasing in Information Centric Networks*. IEEE Global Communications Conference, GLOBECOM 2013, Atlanta, GA, USA, December 9-13, 2013.

Workshops

1. M. Mangili, F. Martignon, A. Capone. *A Network-Planning Perspective for the Migration to Content-Centric Networking*. 16ème conférence ROADEF, Société Française de Recherche Opérationnelle er Aide à al Décision. Marseille, France, February 25-27, 2015.
2. M. Mangili, F. Martignon, A. Capone. *Network Functions Virtualization for Content Delivery Networks: A Network Planning Perspective*. 12th Italian Networking Workshop, Cavalese, Italy. January 14-16, 2015.
3. M. Mangili, F. Martignon, A. Capone. *Efficient Network Content Distribution: Analyzing Caching Performance of CCN and CDN Architectures*. 11th Italian Networking Workshop, Cortina, Italy. January 15-17, 2014.
4. M. Mangili, F. Martignon, S. Paris, A. Capone. *Improving Bandwidth and Cache Sharing in Content-Centric Networks*. 10th Italian Networking Workshop, Bormio, Italy. January 9-11, 2013.

Posters

1. A. Araldo, M. Mangili, F. Martignon, D. Rossi. *Analysis and Design of Next Generation Information Centric Networks*. Forum STIC Paris-Saclay, ENSTA ParisTech, Palaiseau, France. December 2nd, 2014.
2. A. Araldo, M. Mangili, F. Martignon, D. Rossi. *Performance Analysis of Secure, Next Generation Content-Centric Networks*. Forum STIC Paris-Saclay, ENSTA ParisTech, Palaiseau, France. November 13th, 2013.

Part I

Background

CHAPTER 2

Named-Data Networking (NDN)

Named-Data Networking (NDN) [13] is one of the projects funded by the U.S. National Science Foundation under the Future Internet Architecture Program, and it builds on top of the Content-Centric Networking (CCN) project [5, 100], presented for the first time by Van Jacobson in 2006 [97].

Named-Data Networking will be used throughout this thesis as a model for an Information-Centric Network (ICN) and, for this reason, Sec. 2.1 gives a broad presentation of the NDN design. The state of the art regarding security of NDN is presented in Sec. 2.2, while in Sec. 2.3 we discuss transport, routing and fragmentation. Mobility properties are illustrated in Sec. 2.4, and we conclude this chapter by presenting in Sec. 2.5 some examples of NDN applications.

2.1 The NDN Paradigm

In this section we illustrate the NDN paradigm, in particular in Sec. 2.1.1 we describe its general architecture including Interest and Data packets, and the pull-driven communication model. In Sec. 2.1.2, the node model is introduced, while in Sec. 2.1.3 we extensively discuss NDN names and common conventions. Lastly, in Sec 2.1.4 we discuss caching policies for CCN.

Despite the fact that we illustrate the main characteristics of NDN with a certain degree of detail, an in-depth analysis of the NDN features can be found in [5, 9, 13, 100, 168].

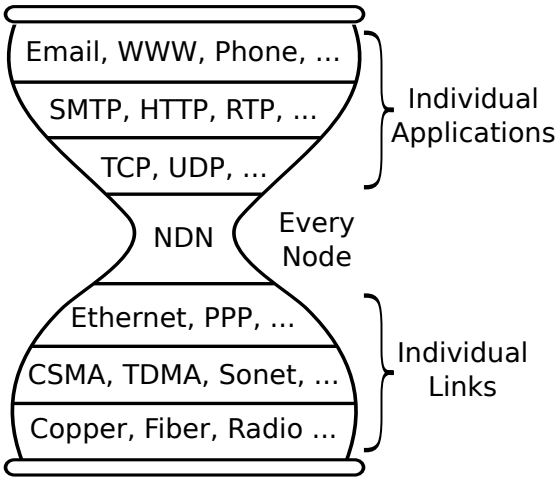


Figure 2.1: *The protocol hourglass and the NDN narrow-waist. NDN is the only protocol that requires to be universally supported by every node in the network, whereas the other protocols require bilateral agreements between individual applications or links.*

2.1.1 Architecture

The objective of NDN is to *"change the semantics of the network service from delivering the packets to a given destination address, to fetching data identified by a given name"* [13]. In order to reach this goal, NDN proposes an alternative protocol for the *thin-waist* of the hourglass model that ensures global connectivity, while providing only the minimal set of functionalities needed for such purpose, similarly to what the universal network layer of IP is currently doing nowadays in Internet.

To mimic the role of IP in today's Internet, NDN is specifically conceived as a universal overlay: not only it can be used on top of any protocol that forwards datagrams (such as IP, TCP, Ethernet, WiFi, etc.), but it also lets any other protocol run on top of it. In this way, as shown in Fig. 2.1, most of the layers in the protocol stack will still require only bilateral agreements, whereas the unique layer demanding universal consensus will still be the network layer (the thin-waist of the hourglass).

As depicted in Fig. 2.2, in NDN there are two types of packets: 1. Interest and 2. Data packets. Demands govern the interactions between nodes in the network, therefore the communication model is *pull*-driven. A node sending *Interests* acts as a content *consumer*, whereas a node that can provide *Data packets* behaves as a *producer*. Routers have a twofold responsibility: 1. they perform packet forwarding and 2. they can also behave as distributed caches.

In NDN, network interfaces are abstracted using the concept of "*faces*". A "face" can either be a physical network interface, but it can also be an application operating on a given node.

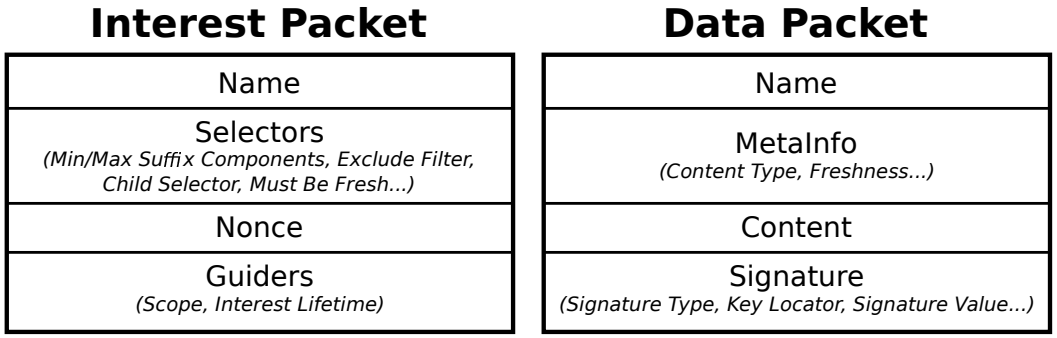


Figure 2.2: *Interest and Data packets in NDN.*

2.1.2 The Node Model

Each node in NDN performs packet forwarding by relying on the presence of three data structures:

1. The Pending Interest Table (PIT);
2. The Content Store (CS);
3. The Forwarding Information Base (FIB).

The PIT is responsible for tracking the list of Interests previously forwarded, but not yet answered. This table stores the *faces* from which Interests were originally received, to implement *reverse path forwarding*: as soon as a router receives a Data packet, it 1. checks the PIT, 2. forwards the packet on the same faces from which Interests for that object arrived (if any) and 3. deletes the corresponding PIT entry.

The content store (CS) acts as a persistent caching storage for the node, and is used to implement universal in-network caching. When an Interest arrives, the router will initially query the CS; in case of a cache hit, the router will be able to directly serve the Data packet. Otherwise, in case of a cache miss, the node will check whether the PIT contains an entry matching the name of the requested content. If such an entry exists, the Interest packet will be dropped, and the face from which the Interest was received will be appended to the PIT entry. On the contrary, if there are no PIT entries for that content, the node will 1. add a new PIT record for the received Interest and 2. forward the Interest towards the data producer. In particular, the next-hop will be chosen by the router's Forwarding Strategy Layer, according to the data in the FIB.

The Forwarding Strategy Layer is responsible for choosing whether the Interest packet should be forwarded upstream, and on which network interface it must be sent. It is also possible in NDN to concurrently forward the same Interest on multiple faces (multicast), and this choice will be performed by the Forwarding Strategy Layer. On top of that, in certain situations, the Strategy Layer may also choose to drop an Interest packet (e.g.: when the upstream link is congested, or for security reasons). In these cases the router can also respond to the Interest with a NACK, in order to signal to the neighbors that a given Interest was dropped.

Due to the soft state persisted in the PIT, Data packets are forwarded on the reverse-path with respect to the Interests, and this symmetry removes the need to provide any data regarding the location of the hosts. For similar reasons, Data packets cannot loop

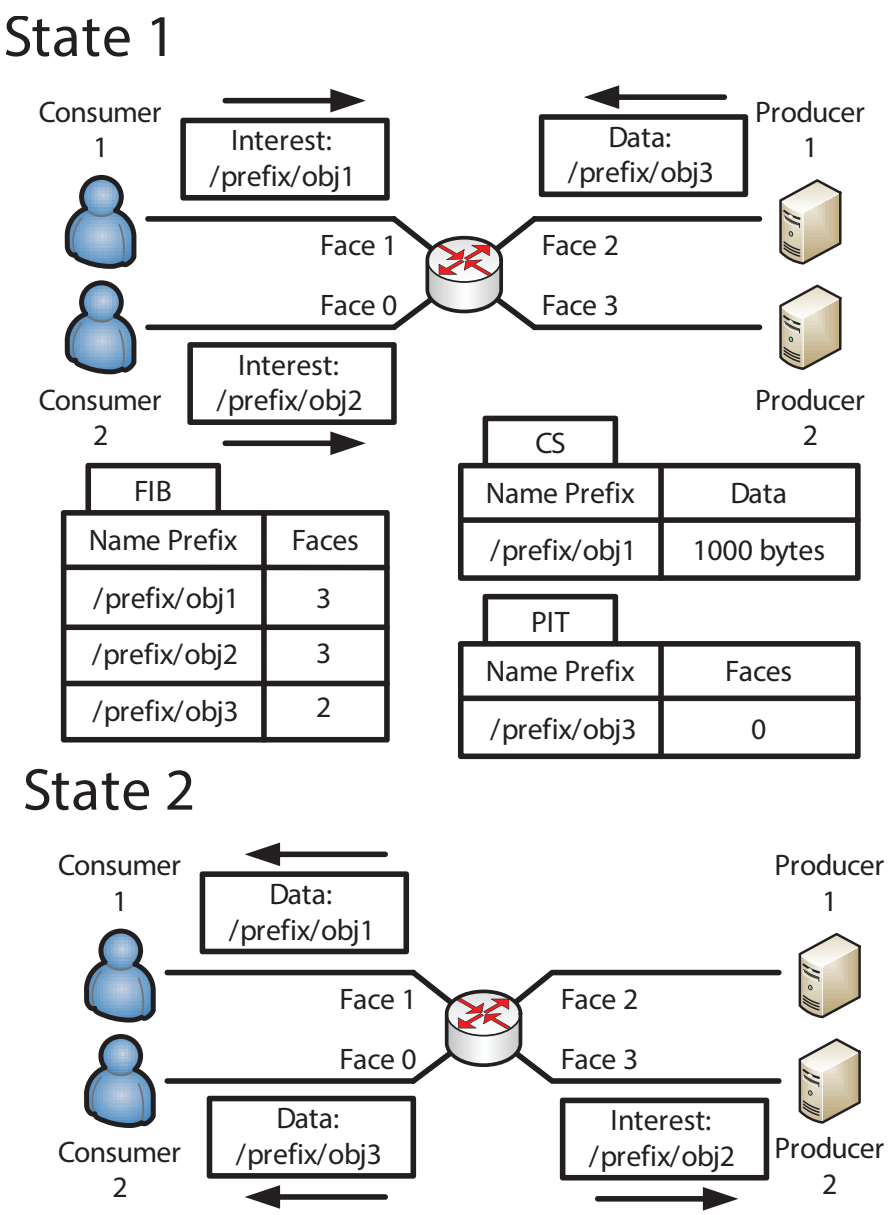


Figure 2.3: *Example illustrating the behavior of a CCN router. Two Interests and one Data packet are received by the router in* State 1. *Given the information contained in the* Pending Interests Table *(PIT), the* Content Store *(CS) and the* Forwarding Information Base *(FIB), in* State 2 *the router forwards two Data packets and one Interest.*

in NDN, whereas Interest loops are prevented thanks to the PIT state. More in detail, each Interest contains a randomly generated Nonce which, coupled with the Name, uniquely identifies the packet. Since both the Interest name and the Nonce are persisted in the PIT, nodes will detect and discard looping Interest packets. Having to deal with soft state in the PIT is one of the major differences of NDN with respect to the stateless IP data-plane.

An example showing the behavior of a CCN router is depicted in Fig. 2.3. In *State 1*,

the router receives two Interests and one Data packet. As shown in Fig. 2.3, the Interest for object `/prefix/obj1` is directly served by the router since it is available in the CS. The Interest for `/prefix/obj2` will be forwarded to *Face 3* since it is the destination available in the FIB [1]. Lastly, the Data packet for `/prefix/obj3` will be forwarded to *Face 0*, as written in the PIT. In *State 2*, the transition to the next state is represented: as described, the router forwards one Interest and two Data packets.

2.1.3 Naming

From the syntactical perspective, NDN proposes a hierarchical structure for content names: they are composed of a sequence of variable-length *components*, each of which is separated by the "/" symbol. At the same time, NDN does not impose any component semantic: the chosen namespace is completely transparent to the network, thus it is up to every application to determine the best namespace conventions according to the type of service that it is going to deliver. The hierarchical naming scheme is a key design choice in NDN that was specifically made to foster scaling of the forwarding and routing functions by means of leveraging prefix aggregation.

In NDN a Data packet satisfies an Interest if the name in the Interest is a prefix for the name in the Data packet.

NDN forces Data packets to be immutable: once a Data packet has been published under a given name, the packet cannot be changed. This constraint is specifically imposed to make sure that universal in-network caching always returns the expected data packet. For this reason, whenever an application needs to publish a new version of the same data, a new name must be generated. NDN also supports content segmentation: if a large data object must be published in the network, it will likely be segmented in a sequence of Data packets. To handle both *versioning* and *segmentation*, the common practice in NDN is to append two additional components to the end of an NDN name [136].

As an example, consider the following name:
`/organization.com/arch/Work.zip/5/18`, where `organization.com` is a globally routable name, `arch/Work.zip` is the object name, whereas the version (i.e., `5`) and the segment number (i.e., `18`) are appended as the two final components of the name, respectively.

In case that a content name is partially known, users can leverage *selectors* in order to retrieve the desired data through one or more interactions. For example, NDN selectors can be used to request the *leftmost* (or *rightmost*) child, given a known name prefix. In this case, by issuing an Interest for `/organization.com/arch/Work.zip/5` and using the *rightmost child* selector, the consumer can retrieve the Data packet of the last segment of `Work.zip`, version `5`. In particular, this feature is possible because the NDN specification defines a canonical ordering for NDN names and components.

Another selector frequently used is the *Exclude filter*: the consumer may send an Interest for `/organization.com/arch` excluding `Work.zip`, to retrieve any other

1. For the sake of clarity, in Fig. 2.3 we omitted the Nonce values. Further details can be found in [13, 193].

content published under `/organization.com/arch`. Furthermore, the NDN specification contains also other selectors as well as *Guiders* for setting further conditions on Interest forwarding; the interested reader may refer to [9] for further details.

2.1.4 Caching Policies

In order to optimize the content distribution performance of NDN, the research community has mostly focused on evaluating the perceived performance of using different caching policies [44, 152, 158, 194]. In particular, caching policies can either be *eviction* or *decision* policies [154, 194].

The *eviction* policy defines the algorithms used to select which stale object should be removed from a full cache, whenever extra space is necessary to store a new content. The most common cache eviction policies are Least Recently Used (LRU), Least Frequently Used (LFU), and Random. On the other hand the *decision* policy is the algorithm that chooses whether a new content should be copied in the cache or not. Decision policies usually have a network-wide effect, and the most common are LCE (Leave a Copy Everywhere), LCD (Leave a Copy Down) or Probabilistic Caching [145, 158].

Remarkable effort has been devoted in finding mathematical models to quantitatively evaluate the performance benefits that caching in NDN can lead to [69, 76, 77, 79, 153]. However, these models usually impose strict conditions on the distribution of input traffic requests, the network topology, as well as the caching policies used.

Whenever these assumptions do not hold, simulators should instead be preferred to mathematical models, and in the literature there are some network simulators that have been specifically designed to assess the performance of the ICN infrastructures [15, 49, 158]. The different simulators are built on top of different frameworks and abstractions; a preliminary comparison of them is provided by Tortelli et al. in [174].

While significant effort has been posed in formulating and evaluating novel caching techniques, only minor attention has been devoted to the network planning problem of the NDN. For this reason, the contribution of this Ph.D. thesis is to tackle the network planning problem of an ICN, to foster efficient content distribution. In particular we focus on different aspects of the network optimization problem: in Chapter 4 we tackle the migration step to NDN, whereas in Chapter 5 we compare the best performance bounds that can be achieved in NDN, with those that can be observed in a CDN. Finally, while still considering the general problem of efficient content distribution, in Chapter 6 we formulate a network planning proposal for a virtual content delivery service based on the novel paradigm of NFV.

2.2 Security

The security model supported by NDN is radically different with respect to the one we use in IP nowadays: rather than creating an encrypted end-to-end connection between two machines, NDN secures the content itself. In other words, NDN lets users place trust in the packets, rather than in the host from which the data was retrieved.

In this section we are going to discuss relevant security properties of the NDN

architecture, in particular in Sec. 2.2.1 we discuss topics related to integrity and trust, Sec. 2.2.2 presents potential risks related to Denial of Service, while privacy issues are investigated in Sec. 2.2.3. Finally, the problem of access control is illustrated in Sec. 2.2.4.

2.2.1 Integrity and Trust

The NDN content-based security model was specifically designed due to the fact that any node in the network can potentially respond to an Interest by providing a cached copy of the requested content. Therefore, NDN forces producers to cryptographically sign each Data packet, in order to let consumers check the received data. More in detail, in NDN content consumers should be able to verify the following properties [168]:

1. *Validity*: the content integrity has not been altered;
2. *Provenance*: the content has been published by the legitimate producer;
3. *Relevance*: the content satisfies the request originally sent by the consumer.

In order to let consumers check these three properties, NDN creates a binding between the name of a content and the corresponding bits of data. Let P be a publisher and let N be the name P has chosen for content C; the NDN Data packet is as follows $M = (N, C, Sign_P(N, C))$, where $Sign_P(N, C)$ is the signature made with P's private key of the digest of (N, C). Interest packets are not signed: as shown in Fig. 2.2, they do not possess the *"Signature"* field.

The Data packet can contain a pointer in the field "Key-Locator", to the NDN name of the public key that can be used to verify the signature of the packet. However, by publishing a public key under a given NDN namespace, producers are implicitly creating a digital certificate for the key. This NDN feature is used to create a novel notion of *trust*: while nowadays it is a common practice to trust a content producer entirely, regardless of the specific content it is providing, the trust model enforced by NDN goes far beyond this limitation and is flexible enough to express very *fine-grained* trust relations.

A proposal for a key distribution architecture was recently formulated by Mahadevan et al. in [124], where a Key Resolution Service (KRS) for NDN/CCN is presented. Inspired by the hierarchical structure of DNS, the proposed KRS system can be used to register, store and efficiently distribute keying materials associated with CCN namespaces, providing adequate support to help consumers check content integrity. Despite the fact that content integrity can be verified by nodes in the CCN, the presence of the distributed caches and the way consumers issue interest packets on the network let adversaries perform *content poisoning* attacks to distribute fake copies of data. Ghali et al. present in [83] a ranking algorithm for cached content, specifically designed to foster eviction from the caching storage of poisoned contents, by gathering statistics on consumers' actions as a result of previous content delivery.

2.2.2 Denial of Service

Additional security implications of this novel network model have also been studied in the literature [14, 82]. In [82], Gasti et al. do a preliminary evaluation of Denial of Service (DoS) attacks in NDN; in particular, they argue that interest flooding as well as content/cache poisoning represent the most serious threats.

Performing such a kind of attacks is demonstrated to be possible in [53], where Compagno et al. present "Poseidon", a framework that uses local detection as well as collaborative techniques based on the push-back approach to limit the feasibility of interest flooding attacks in NDN. DoS attacks are again the topic addressed by Goergen et al. in [86], where they formulate a proposal for a monitoring architecture that can detect attack patterns by tracking recent activity happening over the basic data structure of an NDN node.

While content poisoning attacks aim at disrupting integrity by making adversaries provide novel copies of contents that do not correctly represent the real data requested, cache pollution attacks instead aim at disrupting cache efficiency by breaking the temporal correlation of requests. Since they can severely affect the overall network performance, we group them within the DoS class. As thoroughly discussed in [187], cache pollution attacks can be effectively performed even by an adversary that possesses limited resources. Since cache pollution attacks can have a network-wide effect and are not confined to a single node, Xie et al. present in [187] CacheShield, a pro-active mechanism to prevent cache pollution in CCN. An alternative approach is proposed by Conti et al. in [54], where the authors formulate a novel pollution detection algorithm. Compared to CacheShield, the solution envisioned by Conti et al. has a smaller computational footprint while ensuring at the same time a high overall hit-rate.

2.2.3 Privacy

Despite that Interest packets do not contain the identity of the sender, this feature by itself is not sufficient to ensure better privacy properties than the IP counterpart.

As a matter of fact, both Interest and Data packets leak the name of the contents that users are willing to retrieve, therefore posing remarkable privacy concerns for NDN.

In order to mitigate this issue, DiBenedetto et al. in [60] propose ANDaNA, an adaptation of onion-routing [61] specifically tailored for NDN. By using multiple layers of encryption, they make sure that an eavesdropper cannot leak sensitive information. Another approach based on homomorphic encryption is instead proposed by Fotiu et al. in [74]. Despite the fact that one such proposal is specifically tailored for the Publish-Subscribe paradigm, the authors claim that it can also be applied to NDN and other infrastructures. They envision the implementation of a brokering system that consumers query to retrieve pointers to the requested information items, without making the content provider or the brokers know what type of content the user is willing to access.

Another class of attacks is presented in [11], where Acs et al. focus on timing attacks on caches: in this case, the adversary learns whether a consumer has requested a given content by querying the neighboring routers and analyzing the delay. Cache

privacy techniques are then analyzed with the precise aim to mitigate the effects of a timing attack on the distributed caches. Finally, a comprehensive analysis of privacy issues in ICN, attack categories, adversary models and potential remediation is provided in [71]. More specifically, authors develop a generic ICN model that can be applied to different ICN architectures, to map architectural design choices to potential privacy issues that arise in one such context.

2.2.4 Access Control

The topic of access control in NDN, as clearly recognized in [86], is extremely important and still deserves further attention.

In particular, due to the presence of the distributed caches, any network node can directly respond to an Interest packet, without letting the producer know that one such request has been served. Therefore it is important to make sure that only the subset of authorized users can actually get access to the data, and this requirement must be enforced through the usage of content encryption.

In [201], Zhu et al. propose a mechanism to enforce confidentiality for a conference tool in NDN. Their proposal is based on a centralized approach where the user hosting the conference also handles the generation of symmetric encryption keys to secure the voice stream, while using public-key cryptography to distribute keying materials. Burke et al. formulate in [31] a proposal to secure light control; however their work, like [201], can hardly be extended to other more general cases.

A thorough description of the original encrypted access control mechanism implemented in the CCNx prototype [5] is described by Smetters et al. in [167].

Zhang et al. propose in [195] to enforce the confidentiality and integrity requirements by leveraging the services provided by Identity-Based Encryption (IBE). The main advantage gained by using such technique is that the sender does not need to obtain the public key certificate of the receiver to transmit data securely.

Proxy Re-Encryption (PRE) is used in [186] to support *access-control* in a content-centric network. Among the advantages of one such solution, the overall efficiency of the network is supported by the fact that in-network caching mechanisms can take advantage of this paradigm.

Inspired by the works on secure video content streaming, Misra et al. present in [133] a security architecture specifically tailored to support content distribution in an ICN. By using Broadcast Encryption and, more specifically, a public-key traitor tracing variant of Shamir's threshold secret sharing scheme, their proposed security architecture provides support for confidential communications in CCN, while offering viable techniques to revoke access to any given user. On top of that, another positive advantage of this architecture is that the chosen cryptosystem provides traitor-tracing features, meaning that colluding users disclosing their private decryption keys will indirectly disclose information on their identity.

In [115] Kurihara et al. present CCN-AC, an encryption-based framework for CCN to implement any access control scheme. By leveraging CCN 1.0 manifests, they show that CCN-AC supports group based, as well as broadcast access control.

The solutions reviewed so far promise to provide application-layer services to cope with the access control requirement in NDN, by means of leveraging computationally expensive public-key encryption techniques. In order to reduce the computational overhead introduced on the end-points, they usually leverage symmetric encryption to securely distribute the encrypted data, while public-key encryption is only used to exchange keying materials. However, these approaches are vulnerable to the scenario where colluding consumers disclose their symmetric decryption keys, letting other users to completely by-pass the access control mechanism. On top of that, the presence of the distributed caches often makes it very complex for the producer to get detailed feedback on the set of contents that users have retrieved from the network. Lastly, the reviewed mechanisms do not explicitly take into consideration the presence of the distributed caches in NDN. In Chapter 7 we tackle all these challenging issues and formulate ConfTrack-CCN, a security mechanism based on standard encryption techniques that can support Access Control functionalities on NDN while being robust with respect to the above-mentioned issues.

2.3 Transport, Routing, Fragmentation

In this section we consider three important problems in NDN: in Sec. 2.3.1 we present the most recent guidelines to implement transport-layer services for NDN. Routing is discussed in Sec. 2.3.2, while in Sec. 2.3.3 the topic of Fragmentation is presented.

2.3.1 Transport

The current specification for NDN delegates the functionalities enforced by the transport layer to the application themselves, through the usage of external support libraries. More in depth, while multiplexing/demultiplexing are natively supported (thanks to the usage of Faces), reliable delivery and congestion control are instead out of the scope of the functionalities provided by NDN, and need external support.

In the literature different designs for reliable delivery and congestion control have been formulated [35–37, 193] and two common rationales are shared by all these proposals: 1. they leverage the stateful NDN data-plane by using the data available in the PIT and 2. they foster Interest packets re-submissions or shaping, in order to enforce reliable packet delivery and congestion control, respectively.

A preliminary work by Yi et al. [193] studies the role of the Strategy Layer to support adaptive packet forwarding. The point-to-multipoint nature of NDN fosters the usage of advanced forwarding strategies, in particular the authors suggest to rank candidate outgoing interfaces in the FIB, according to a given performance metric (e.g., by increasing values of the RTT). On top of that, they also support the usage of NACKs: whenever the lifetime of an entry in the PIT table expires, a destination is unreachable, or congestion is detected, the router may send a NACK packet back to the Faces from which the given Interest was originally received. Finally, the authors suggest to leverage an Interest rate limit in order to maintain the one-to-one flow balance between Interests

and Data packets.

A pioneering work on congestion control strategies for NDN has been provided by Carofiglio et al. in [35–37]. In particular the authors formulate in [35] a receiver-driven window-based interest control protocol (ICP), that regulates the Interest packet rate with an AIMD mechanism, showing that one such mechanism achieves optimal max-min fairness even in the presence of the distributed caches. They also extended their original proposal to the multipath scenario in [37], and finally showed in [36] that by using a hop-by-hop ICP, faster congestion avoidance and rate adaptation can be achieved, while still guaranteeing the fairness and efficiency properties.

2.3.2 Routing

The content-based addressing of NDN, and its forwarding model are a strict superset of IP, therefore any routing protocol that works well in IP should also support NDN out of the box [100]. In [192] Yi et al. show that having the intelligent forwarding layer provided by NDN makes routing protocols easier by no longer requiring to explicitly handle short-term churn.

A named-data link state inter-domain routing protocol (NLSR) is presented by Hoque et al. in [92]. NLSR extends OSPFN [181] to foster point-to-multipoint communications in NDN, by providing multiple paths for the same name prefix. On top of that, it leverages the NDN pull-based communication model as well as its provided security functionalities, to securely distribute Link State Advertisements (LSAs). NLSR not only shows that it is fairly straightforward to adapt IP routing protocols to the NDN naming scheme, but it also provides useful insights on best practices to build new NDN applications, especially regarding the design of the content naming scheme, trust management and key distribution.

Multiple paths to the same name prefix are computed by a router in NLSR by finding the shortest path towards a given content and then iterating the same computation by removing the adjacent link belonging to the previous selected paths. To go beyond the inefficiencies introduced by one such an approach, Garcia-Luna-Acevez in [80] proposes an efficient distance-based content routing protocol (DCR) for ICN. Compared to traditional distance-vector (DVR) and link-state routing (LSR), DCR requires less storage and computation due to the fact that the network topology is not propagated and updates only require nearest-prefix copies.

2.3.3 Fragmentation

Lastly, we would like to point out that hop-by-hop packet fragmentation and end-to-end reassembly (as in the current version of IPv4) is not a viable choice for NDN, for the following reasons: 1. partial Interest packets cannot be processed by intermediate nodes, 2. a full Data packet is required in order to perform a matching with a corresponding Interest packet and 3. caches should have the entire data packet, in order to serve heterogeneous requests coming from clients with a different MTU. Furthermore, an end-to-end fragmentation and reassembly is not viable either, because the value of the path MTU is content-dependent and can potentially change as a result of data being

cached in intermediate nodes. For all these reasons, current guidelines for NDN require hop-by-hop packet fragmentation and reassembly [16].

2.4 Mobility

The design of Named-Data-Networking natively overcomes many of the limitations of IP to support communications in a highly mobile and topological dynamic environment [200].

First of all, by removing host-based addresses, NDN permits to avoid the extra-costs related to the acquisition of a new IP whenever a node connects to a new network. Moreover, the connection-less approach and the content-based security jointly make it possible to seamlessly handle consumers mobility: if a node migrates to another network it is sufficient to make the node re-issue potential unsatisfied Interest packets. For similar reasons, also host multihoming can easily be supported [175].

Intermittent connectivity can take advantage of the presence of the distributed caches, which foster data distribution in a similar way as done by Delay Tolerant Networks (DTNs) [98]. For example, Han et al. propose in [89] AMVS-NDN, an adaptive video streaming solution, tailored for NDN, that can support the presence of the distributed caches for the efficient delivery of video content.

On top of all these advantages, NDN can also better exploit the broadcast nature of the wireless medium: nodes can directly respond to Interests packets if they possess a copy of the requested content, moreover they can also overhear data packets and populate their content caches with popular contents without paying additional retransmission costs [129]. By accurately designing the forwarding protocol, as done in [130], NDN can be used to efficiently forward data in a multi-hop, highly dynamic wireless network. In [87] Grassi et al. show that NDN is an interesting solution to support vehicular network communications. In fact, while smartphones and portable devices are usually quite constrained in terms of available caching storage, as well as power consumption, vehicles, on the contrary, do not have those constraints and can potentially be used to physically move very large quantity of data from one location to another. For these reasons, NDN is very appealing to be used in VANETS.

However, while NDN can seamlessly handle consumers' mobility, producers' mobility is instead more challenging [18, 196], and yet, dealing with producers' mobility is an indispensable feature in NDN due to its pull-driven communication model. As a matter of fact, whenever a mobile client wants to upload data to a remote node, it will behave as a *content producer*: the server must issue Interest packets to retrieve the remote content provided by the mobile client. To support producers' mobility in [196] Zhang et al. propose to leverage the stateful NDN data-plane functionalities and use periodic Interests packets sent by the mobile producer towards a routable anchor in order to disseminate PIT entries in the network. This data is then used as breadcrumbs to correctly forward incoming Interest packets towards the location of the mobile producer. Finally, Attam et al. show in [21] that NDN can easily be used on top of a Bluetooth connection.

In Sec. 3, we study the content distribution problem on a wireless scenario in NDN.

In particular, we leverage the consumers' mobility services provided by NDN to foster the efficient fruition of digital contents provided by a producer. In our vision the content provider leases unused bandwidth and caching storage available in residential access points (APs). Economic incentives will be provided to the AP owners who chose to participate to the mechanism.

2.5 Applications

NDN is designed and engineered as a *network-layer* protocol, that can be used regardless of the specific application scenario. However, to foster the migration from a host-based to a content-centric network paradigm, it is vital to have adequate support for NDN from the applications themselves, in order to take advantage of all the positive benefits that users may experience thanks to this new network protocol. In particular, in this section we present some of the applications built on top of NDN.

In this context, some preliminary works have considered different aspects related to the web experience: in [146], the authors present an extension of WebKit for a browser capable to retrieve NDN/CCN objects, and in [165] the NDN.JS JavaScript library is presented, to provide primitives to retrieve NDN contents through WebSocket requests. On the server-side, two approaches have been considered: in [184] and [32] different authors present an HTTP/CCN gateway, while in [147] Qiao et al. describe a CCN-native implementation of the popular Tomcat application server.

In most of these solutions, as well as in other works, standard application-layer protocols such as HTTP and WebSockets are used on the endpoints [32, 164, 165, 184, 190]. For instance, in [184] an HTTP-CCN gateway is presented: according to their design, both the client and server support HTTP, while special intermediate nodes, known as the Ingress and Egress gateways, are responsible to translate the incoming HTTP requests to NDN/CCN packets. Similarly, in [165], the client node connects through WebSocket to a Proxy which extracts the original NDN packet and forwards it to a default CCN router via TCP or UDP. The main advantage by using this technique is that only few modifications must be implemented on intermediate nodes, while the endpoints do not have to explicitly implement the content-centric protocol stack.

An alternative approach advocates to directly use NDN/CCN packets as the enabling protocol for the web [146, 147, 165]. More in detail, Shang et al. describe in [165] a Firefox plugin based on the XPCOM interface, that let users browse a new "`ndn:`" namespace using raw TCP or UDP sockets; similarly Qiao et al. envision a scenario in which a native NDNBrowser based on WebKit is used [146], while in [147] the same authors implement an NDN-Tomcat application server. To foster one such paradigm shift, in [146, 147] both push and pull communication models are considered, segmentation issues are discussed, as well as problems related to both caching of static contents, and preliminary support for dynamic pages. While being extremely interesting and innovative approaches, we believe that NDN/CCN should be considered as a network-layer protocol, whereas, at the application layer level, we would expect to see other protocols and abstractions, more suited to support the typical web interactions. As a matter of fact, while the proposals in [146, 147] provide custom responses

to mimic an HTTP-like interaction in which the client can send data through a GET or POST HTTP method, they do not entail the full set of functionalities supported by an application layer protocol such as HTTP.

Still in the web domain, an orthogonal issue that we think will be very important for the practical application of the content centric paradigm is to take into consideration its effects on the way web developers conceive and code dynamic web applications: the requirement to deal with immutable content objects, the necessity to support in-network caching, as well as the impact on the security requirements (as discussed in Chapter 7), will likely have a major impact on the way content-centric applications are conceived.

Other applications have been implemented using NDN/CCN: in [99] VoCCN (Voice-over Content Centric Networks) is described. In particular, VoCCN is implemented by encapsulating standard VoIP protocols (SIP, SRTP) in CCN packets; the architecture of the prototype is described in great detail to provide an example on how it is possible to adapt current applications to the novel CCN/NDN layer. The content-based paradigm dramatically ease the development of distributed services and applications, as demonstrated in ChronoChat, a decentralized chat built on top of the NDN ChronoSync service [199], as well as in ACT, a decentralized audio conference tool for NDN [202]. These examples demonstrate that NDN/CCN can be adopted as robust solutions to implement the online text and voice chat experience, while making straightforward to implement these services in a distributed way.

A number of prototypes for video distribution in NDN have emerged [114, 183, 202]: by taking advantage of the distributed caching both live streaming and on-demand video distribution can benefit of the content-centric paradigm. Finally, other relevant CCN/NDN applications are listed in [3, 8].

Part II

Wireless NDN

CHAPTER 3

Bandwidth and Cache Leasing in Wireless NDN

In this chapter we consider the problem of content distribution in a wireless Named-Data Network: mobile clients can connect to wireless access points owned by third parties and download digital contents published by a content provider. In this heterogeneous network scenario, the bottleneck link is often the backhaul Internet connection of the access point. In order to efficiently exploit the WiFi channel, while offloading the origin server of the content provider, in this chapter we propose to apply the NDN paradigm to wireless communications, to foster the fruition of digital contents in mobility.

In particular, we propose a strategy to stimulate third parties to jointly lease the unused bandwidth and storage available on wireless access points in a heterogeneous Named-Data Network. We formulate this problem as a combinatorial reverse auction run by a content provider willing to increase the number of users reached by his service. While serving a larger number of users, the presence of the distributed caches let the provider reduce the costs spent to run the distribution infrastructure, while jointly saving energy.

First of all we show that the optimal allocation with partial coverage problem is NP-hard, we then provide greedy heuristics that guarantee the *individual rationality* and *truthfulness* properties, and compare their performance numerically. We evaluate the benefits of our proposed mechanisms in terms of the cost savings for the content provider obtained by offloading his infrastructure through the caches, as well as the reduced computational time necessary to execute the allocation algorithms.

Finally, we further analyze a fully distributed approach where mobile clients au-

tonomously select the access point to use, and we model this scenario as a congestion game, showing that it exhibits desired properties (i.e., existence and uniqueness of a Nash Equilibrium).

3.1 Introduction

In recent years, the worldwide success of smartphones and tablets, and the adoption of LTE for broadband wireless connections have promoted the fruition of popular online services, in mobility [143]. More in detail, recent analysis on data traffic for mobile access networks have shown that, during peak period, 40% of the downstream mobile traffic in North America is accountable to video contents, whereas a remarkable 26% is dedicated to social networking. Apps download from popular online stores have also become a remarkable entry, generating more than 6% of the overall data downloaded in mobility [160], and this demand is going to increase even further [171].

Even though the deployment of LTE has dramatically raised the available bandwidth, the average monthly traffic consumption for mobile traffic is still orders of magnitude lower than the one for fixed access, going from 500 Mbytes to almost 50 Gbytes per month, on average, in North America [160]. For this reason, two complementary techniques are proposed as viable solutions to satisfy the ever-increasing bandwidth demand for mobile clients, namely heterogeneous networks and WiFi offloading [47]. In particular, very promising research results, show that nowadays, WiFi connections offload up to 65% of the total mobile data traffic, mostly thanks to the broadband connectivity provided by residential access points [118], whereas by using small cells, the scarce spectrum resources can be used more efficiently [34, 166].

By applying the WiFi sharing model on a global scale, one can also envision the realization of Public Access WiFi Networks (PAWS). A feasibility study for PAWS is presented in [161]. The key findings discovered by the authors show that many domestic broadband connections have enough spare capacity to share the available WiFi connectivity; furthermore they also found out that one such type of service is going to be exploited in many different ways by users, and its adoption will likely be very relevant. Similar conclusions were also envisioned by Landa-Torres et al. in [116], where the authors focus instead on the network planning problem, to maximize the overall signal coverage, under a budget-constrained condition.

In this chapter we analyze a relevant scenario where the content provider has the simultaneous objectives of (1) extending his mobile customer base by offering the users an ubiquitous access to the provided content and (2) saving OPEX and energy costs by performing server offloading, exploiting the built-in caching features of NDN.

Since the content provider should not bear the costs for the realization of a new access network, we propose the creation of a marketplace where *third party access point owners* offer their unexploited bandwidth and storage resources in exchange for economic incentives for their cooperation. However, misbehaving access point owners may jeopardize the efficiency of the allocation mechanism by choosing strategically to declare false valuations for the offered resources. In order to solve this issue, *auction theory* provides insights for the design of tamper-proof mechanisms characterized by

the fact that the dominant strategy of every bidder is to declare the real valuation for the provided resources.

In our vision, the user buys a digital content, and the content provider offers the connectivity to retrieve the corresponding data. As an example, we believe that this *"wireless shopping"* business model can be effectively applied to online content stores such as e-book libraries, music and video streaming services, online magazines and newspapers as well as application stores. Some steps towards this direction have been done by Amazon, which is providing to Kindle users the "Free 3G" mobile broadband connection service to wirelessly browse the store, purchase and download the content, while our proposal is specifically tailored for a heterogeneous NDN [2].

The contributions of this chapter are summarized as follows:

1. We design an optimal mechanism that can be used to motivate access point owners to jointly lease their unused bandwidth capacities and cache storage, in exchange for economic incentives. The mechanism is a *reverse auction* that guarantees both the *individual rationality* and *truthfulness* properties, under partial coverage constraints for the mobile clients, while forcing the access point owners to declare their real valuations for the provided resources.
2. We show that the proposed optimization problem is NP-hard. In order to cope with the computational complexity, we then provide three variants of a greedy algorithm that can be used to obtain a close to optimal solution in polynomial time.
3. To further complement previous results, we analyze the mobile clients access point selection problem and we consider the solution obtained in a distributed scenario by modeling the clients' behavior as a *congestion game*. We prove the game is weighted potential and that the Nash Equilibrium is unique; moreover we provide a lower bound on the Price of Anarchy.
4. We provide performance comparisons of the proposed strategies. We show that all the variants of the greedy algorithm outperform the optimal one in terms of execution time. However, this comes at the cost of a reduced profit for the provider, due to the sub-optimality property of the solution computed by such greedy algorithms.

This chapter is structured as follows: Sec. 3.2 describes the network architecture and motivates our proposal. Sec. 3.3 formulates the optimal combinatorial reverse auction as an optimization model, while Sec. 3.4 proposes three greedy algorithms to solve the allocation problem in polynomial time. The mobile clients' allocation problem is described and analyzed in Sec. 3.5. Numerical results are discussed in Sec. 3.6, while in Sec. 3.7 we survey related work. Finally, concluding remarks are presented in Sec. 3.8.

3.2 Bandwidth and Cache Leasing in NDN

In this section we illustrate the principles, definitions and assumptions characterizing the communication network of our scenario. Sec. 3.2.1 describes the network

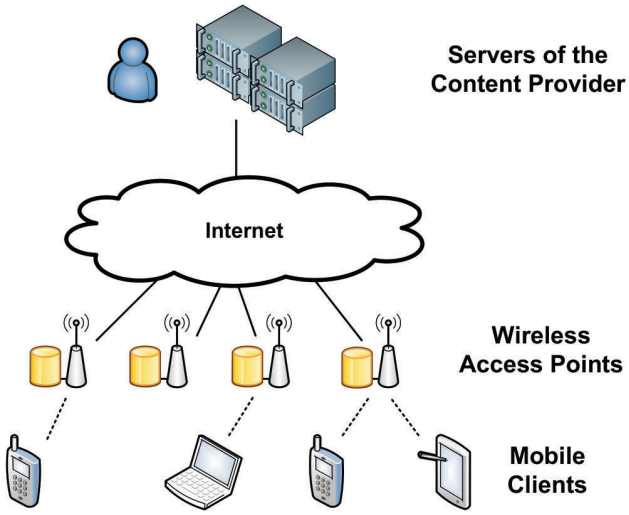


Figure 3.1: Network architecture. *A single content provider (*CP*) is considered in our design. Access points (*APs*) are equipped with caching storage. Mobile clients (*MCs*) connect to a subset of the available* APs*. The demand of a* MC *will be satisfied either by cached content, or by retrieving the data directly from the servers of the* CP *using the available backhaul Internet connection. Our mechanism determines the* MCs *to* APs *allocation that minimizes the total cost.*

architecture and discusses the benefits provided by our allocation algorithm to the content provider, while Sec. 3.2.2 clarifies the structure of the economic incentives to the access point owners and explains the properties enforced by the auction mechanism.

3.2.1 System Model

A graphical representation of the system model we consider in this chapter is shown in Fig. 3.1. A single content provider (CP) wants to lease a set of wireless access points (APs) in order to increase his customer base by providing an ubiquitous content delivery service to mobile clients (MCs).

The core business of the provider is to distribute contents that will be accessed by the customers under payment of a fee. The hardware infrastructure owned by the CP does not comprise the radio access network, since it is beyond the scope of the service it provides; however, we assume that the CP owns a set of content distribution servers reachable through any Internet connection.

Raising the number of customers reached by the service jointly increases the *remuneration* of the CP as well as the *operational costs* (OPEX) due to the increased load of the computational infrastructure. Due to the presence of in-network caching we propose that the operator leases not only the radio access and backhaul connection capacities, but also portions of the caches available at the access points. By exploiting the caching feature of NDN we make the content move across the network towards the locations where most of the users are requesting it, and thus making the provider save a significant amount of the overall energy costs spent to operate the distribution infrastructure.

3.2.2 Economic Incentives

Access point owners can participate to the bandwidth allocation phase by submitting to the CP the bid $[b_j, s_j]$, where b_j is the price at which the j-th AP agrees to jointly share the unexploited backhaul and wireless bandwidth, as well as a quantity s_j of cache. As commonly assumed in the literature (e.g., [93, 111]), in our case the real valuation v_j of each owner is kept hidden; thus, in the most general case, $v_j \neq b_j$.

Let $p_j \in \mathbb{R}^+$ be the price paid by the CP to lease the j-th AP. The utility function of the AP owner is such that:

$$u_j = \begin{cases} p_j - v_j & \text{if AP } j \text{ is selected} \\ 0 & \text{otherwise.} \end{cases} \tag{3.1}$$

We say that *individual rationality* holds if the utility of each player is always non-negative: $\forall j \in \mathcal{A}, u_j \geq 0$.

When the CP has collected all the sealed bids of the APs, he will select a set of access points that should be rewarded, with the aim to maximize his overall revenue. In particular, by serving the traffic demand d_i of a mobile client $i \in \mathcal{M}$, the operator will obtain a profit P per unit of bandwidth. Moreover, a cost C, proportional to the requests that will generate a cache miss at the access points, must also be considered, to account for the energy *charges* experienced by the CP to serve the incoming requests with his infrastructure. Thus, we would like to design a two step mechanism that 1. chooses which wireless access points should be selected among those that participated to the allocation, and 2. computes the rewards paid to the corresponding winners, in such a way that every user is forced to declare a price equal to the true valuation of his offer.

We make the assumption that each access point is directly connected to a backhaul Internet connection, whose capacity is known to the operator. We also assume that it is very unlikely that the AP owner declares a false quantity of available cache storage, since this misbehavior can be easily detected and punished by the content provider. Notwithstanding, the mechanism needs to force the AP owners to declare their real valuations, since they may be tempted to lie in order to increase their utilities by behaving strategically.

3.3 Optimal Allocation and Payment Scheme

This section illustrates the optimal mobile clients allocation and access point payment algorithm. We will derive the optimal allocation rule in terms of: 1. the prices paid by the content provider (CP) to remunerate the selected access point owners, 2. the operational cost (OPEX) faced by the CP due to the load on his computational infrastructure (*energy* costs), and 3. a traffic-proportional profit that the CP experiences for serving the clients' requests. The notation used in this chapter is summarized in Table 3.1.

We denote with $\mathcal{M}$ the set of mobile clients (MCs) in the heterogeneous network, while $\mathcal{A}$ is the set of access points (APs). Each MC $i \in \mathcal{M}$ generates a traffic demand d_i that might be satisfied by at most one AP $j \in \mathcal{A}$.

Due to locality constraints, the APs have a limited maximum coverage radius, that we denote with L_j for AP j. Let $l_{i,j}$ be the distance between MC i and AP j, and

Table 3.1: *Summary of the notation used in this chapter.*

Parameters of the ILP model	
$\mathcal{M}$	Set of mobile clients
$\mathcal{A}$	Set of access points
b_j	Bid of AP j
s_j	Storage space for caching offered by AP j
h_j	Average hit-rate for AP j
R_j	Backhaul bandwidth available at AP j
L_j	Maximum coverage radius of AP j
$l_{i,j}$	Distance between MC i and AP j
$r_{i,j}$	Maximum Wi-Fi rate of the MC i when it is connected to AP j
$q_{i,j}$	MC i, AP j connectivity. $q_{i,j} = 1 \iff r_{i,j} > 0$, otherwise $q_{i,j} = 0$
d_i	Traffic demand of MC i
C	Cache miss cost per unit of bandwidth, paid by the CP
P	Average profit that a unit of bandwidth generates for the CP

Variables of the ILP Model	
$x_{i,j}$	Binary variable that indicates whether MC i is assigned to AP j
y_j	Binary variable that indicates whether AP j is selected or not

Parameters of the Auction	
v_j	Private valuation of AP j
p_j	Actual price paid by the CP to the j-th AP owner
u_j	Utility function of the j-th AP owner

let $r_{i,j}$ be the maximum rate of MC i when connected to AP j. If MC i is beyond the range of AP j, the corresponding rate will be null: $r_{i,j} = 0, \forall i \in \mathcal{M}, j \in \mathcal{A} \mid l_{i,j} > L_j$. For the sake of simplicity, hereafter we assume that the rate $r_{i,j}$ refers only to the maximum possible rate between MC i, and AP j, regardless of the other MCs served by AP j. In Sec. 3.5 we extend this result by studying the behavior of the mobile clients with respect to the congestion they experience by connecting to the AP j.

In order to participate to the auction, AP owners are required to declare s_j, the amount of storage they will offer for caching purposes. This quantity will be used by the content provider (CP) in order to compute the average cache hit rate for AP j, denoted by h_j. If $h_j = 0$ this means that AP j does not cache any object.

Cache misses increase the load on the computational infrastructure of the CP, forcing the provider to spend additional energy costs necessary to provide the requested contents. We denote with C the cost per unit of bandwidth due to cache misses, while P is the profit that the CP will experience for serving one unit of bandwidth of users' requests. Lastly, let R_j be the bandwidth of the backhaul connection available at AP j.

Our formulation admits *partial coverage*, that is, the optimal solution may not serve all the available MCs since, in some cases, this may not be the best choice for the CP. The binary variable $x_{i,j} \in \{0, 1\}$ is used to represent the assignment between MC i and

AP j, and is such that:

$$x_{i,j} = \begin{cases} 1, & \text{if MC } i \text{ is assigned to AP } j \\ 0, & \text{otherwise.} \end{cases} \tag{3.2}$$

Furthermore, we denote with the binary variable y_j the set of APs selected by the auction:

$$y_j = \begin{cases} 1, & \text{if AP } j \text{ is selected by the auction} \\ 0, & \text{otherwise.} \end{cases} \tag{3.3}$$

Given the above definitions and assumptions, the mobile clients allocation problem (MCAP) can be formulated as follows:

$$\max P \sum_{i \in \mathcal{M}} \sum_{j \in \mathcal{A}} x_{i,j} d_i - \sum_{j \in \mathcal{A}} y_j b_j - \sum_{i \in \mathcal{M}} \sum_{j \in \mathcal{A}} x_{i,j} d_i (1 - h_j) C \tag{3.4}$$

s.t.

$$\sum_{j \in A} x_{i,j} \leq 1 \qquad \forall i \in \mathcal{M} \quad (3.5)$$

$$\sum_{i \in \mathcal{M}} \frac{d_i x_{i,j}}{r_{i,j}} \leq 1 \qquad \forall j \in \mathcal{A} \quad (3.6)$$

$$\sum_{i \in \mathcal{M}} d_i x_{i,j} (1 - h_j) \leq R_j \qquad \forall j \in \mathcal{A} \quad (3.7)$$

$$x_{i,j} \leq y_j \qquad \forall i \in \mathcal{M}, j \in \mathcal{A} \quad (3.8)$$

$$x_{i,j} \in \{0, 1\} \qquad \forall i \in \mathcal{M}, j \in \mathcal{A} \quad (3.9)$$

$$y_j \in \{0, 1\} \qquad \forall j \in A. \quad (3.10)$$

The objective function (3.4) maximizes the total revenue of the content provider, which is given by: I) the traffic-proportional profit the CP is going to obtain for serving MCs' demand: $P \sum_{i \in \mathcal{M}} \sum_{j \in \mathcal{A}} x_{i,j} d_i$; II) the incentives paid to remunerate the selected access points for their storage and access bandwidth: $\sum_{j \in \mathcal{A}} y_j b_j$; and III) the infrastructure costs caused by cache misses: $\sum_{i \in \mathcal{M}} \sum_{j \in \mathcal{A}} x_{i,j} d_i (1 - h_j) C$. It is important to note that, as in the standard Vickrey auction [137], we optimize with respect to the user's provided bids b_j rather than the actual price paid to the AP owners. In fact, prices will be determined as a second stage, only after having selected the most promising APs. On top of that, the pricing rule that we are going to use, is such that the *truthfulness* property is guaranteed.

In (3.5) we express the set of *coverage* constraints, which force every mobile client to be assigned to at most one access point. Coverage is *partial*, because there might be cases where it is better not to serve some MCs' since it is not economically viable for the CP (e.g., if there is a single and very expensive AP).

Algorithm 1: Optimal and Truthful Reverse Auction

Input : $\mathcal{M}, \mathcal{A}, \mathbf{b}, \mathbf{s}, \mathbf{d}, \mathbf{r}, \mathbf{h}, \mathbf{R}, C, P$

Output: $\mathbf{y}, \mathbf{p}, \mathbf{x}$

1 $(y_j, x_{ij}) \Leftarrow$ Solve the ILP model ;

foreach $j \in \mathcal{A} : y_j = 1$ **do**

2 $\quad (y_j^{'}, x_{ij}^{'}) \Leftarrow$ Solve the ILP model without AP j ;

3 $\quad p_j \Leftarrow \sum_{i \in \mathcal{M}} \sum_{n \in \mathcal{A}} \left(x_{i,n} - x_{i,n}^{'}\right) d_i \left[P - C\left(1 - h_n\right)\right] - \sum_{\substack{n \in \mathcal{A} \\ n \neq j}} \left(y_n - y_n^{'}\right) b_n$;

end

Constraints (3.6) and (3.7) limit the number of MCs assigned to each AP, given that the radio access network and the backhaul Internet connection have a bounded capacity. Constraints (3.6) impose that the total demand served by an AP does not exceed the capacity of the radio access network; in particular the i-th MC expresses a traffic demand d_i and is served at a rate $r_{i,j}$ when connecting to AP j, thus consuming a fraction $\frac{d_i}{r_{i,j}}$ of the available wireless resources. On the other hand, constraints (3.7) consider the fact that the backhaul Internet connection serves only the aggregate demand that generates a cache miss (i.e., the portion of the MC demand that must be directly served by the CP's servers).

Constraints (3.8) make sure that MCs can associate only to the APs selected by the mechanism. Finally, the sets of constraints (3.9) and (3.10) express the integrality conditions on the decision variables.

After having defined the ILP model used to solve MCAP, we now illustrate the algorithm that forces the AP owners to bid their real valuations. *Algorithm 1* formalizes the steps that the auctioneer should follow in order to determine the APs that should be selected as well as their payments for the provided resources. It returns the list of selected APs $\mathbf{y}$, the assignment matrix of MCs to APs $\mathbf{x}$ and the corresponding payments $\mathbf{p}$.

The algorithm proceeds in three steps. In Step 1, the ILP model (3.4)-(3.10) is solved and the maximum revenue allocation $(y_j, x_{i,j})$ is computed. In Step 2, the solution of the ILP model without the j-th AP is determined, in other terms, the algorithm solves again the ILP model (3.4)-(3.10) with the additional constraint $y_j = 0$. Finally, Step 3 computes the optimal price for the j-th AP according to the Vickrey-Clarke-Groves mechanism with Clarke pivot rule [137]. The optimal price represents the "*opportunity cost*" that the presence of the j-th AP causes to the other bidders.

Hereafter we show that the MCAP problem is NP-hard, by a polynomial time reduction of the Knapsack Problem (KP) to the ILP model (3.4)-(3.10).

Theorem 1. ***Complexity of the MCAP problem.*** *The MCAP problem is NP-hard.*

Proof. In order to prove the theorem we polynomial-time reduce the NP-hard Knapsack Problem (KP) [128] to MCAP.

Let $\mathcal{I}$ be a set of items of cardinality $n = |\mathcal{I}|$; for each item $i \in \mathcal{I}$, let v_i be the value of the item and w_i be its weight. Moreover, the binary variable $x_i \in \{0, 1\}$ is set to 1

if item $i \in \mathcal{I}$ belongs to the subset $I \subseteq \mathcal{I}$, and 0 otherwise. The knapsack problem is to select a subset of items $I \subseteq \mathcal{I}$, such that their total value $\sum_{i \in \mathcal{I}} x_i v_i$ is maximized, and their total weight is bounded by W, that is $\sum_{i \in \mathcal{I}} x_i w_i \leq W$.

We show that every instance of the knapsack problem can be reduced in polynomial time to an instance of the mobile clients allocation problem (MCAP), as follows. First of all, we set $|\mathcal{A}| = 1$, whereas $|\mathcal{M}| = n$. Furthermore, we force $h_1 = 1$, $b_1 = 0$, $d_i = \frac{v_i}{P}$ and $r_{i,1} = \frac{v_i W}{P w_i}$. The constructed MCAP problem is as follows:

$$\max P \sum_{i \in \mathcal{M}} \sum_{j \in \mathcal{A}} x_{i,j} \frac{v_i}{P} - \sum_{j \in \mathcal{A}} y_j 0 - \sum_{i \in \mathcal{M}} \sum_{j \in \mathcal{A}} x_{i,j} d_i (1-1) C \tag{3.11}$$

s.t.

$$x_{i,1} \leq 1 \qquad \forall i \in \mathcal{M} \tag{3.12}$$

$$\sum_{i \in \mathcal{M}} \frac{\frac{v_i}{P} x_{i,1}}{\frac{v_i W}{P w_i}} \leq 1 \tag{3.13}$$

$$\sum_{i \in \mathcal{M}} d_i x_{i,1} (1-1) \leq R_1 \tag{3.14}$$

$$x_{i,1} \leq y_1 \qquad \forall i \in \mathcal{M} \tag{3.15}$$

$$x_{i,j} \in \{0,1\} \qquad \forall i \in \mathcal{M}, j \in \mathcal{A} \tag{3.16}$$

$$y_j \in \{0,1\} \qquad \forall j \in A. \tag{3.17}$$

The constructed instance of MCAP is equivalent to the original instance of KP, and one such construction is done in polynomial-time. Therefore, MCAP is NP-hard. □

Since the objective function we take into account is not the standard used in the VCG scheme, hereafter we prove that our mechanism jointly enforces the *individual rationality* and *truthfulness* properties.

Theorem 2. ***Individual Rationality.*** *The payment rule satisfies the individual rationality property, i.e.:*

$$\forall j \in \mathcal{A} \mid u_j \geq 0. \tag{3.18}$$

Proof. Given the utility function we consider, the AP owners selected by the mechanism have a utility $u_j = p_j - v_j$, whereas the others experience a zero utility. Let $SW(y_j, x_{i,j})$ denote the value of the social welfare (i.e.: the objective function (3.4)). Our payment rule for the selected APs ensures that $p_j = SW(y_j, x_{i,j}) - SW(y_j^{'}, x_{i,j}^{'}) + b_j$. Therefore, when the AP owner declares his valuation truthfully, that is $b_j = v_j$, the utility will be non-negative $u_j \geq 0$, since $SW(y_j, x_{i,j}) \geq SW(y_j^{'}, x_{i,j}^{'})$. □

Theorem 3. ***Truthfulness.*** *The payment rule ensures the truthfulness property, that is, a dominant strategy for all the players is to declare $b_j = v_j$.*

Proof. Let (x, y) and $(\hat{x}, \hat{y})$ be the solutions of MCAP (3.4)-(3.10), when the AP owner declares $b_j = v_j$ and $\hat{b}_j$, $b_j \neq \hat{b}_j$, respectively. Moreover, let (x^{-j}, y^{-j}) be the solution of (3.4)-(3.10) without considering the AP j. The utility of j when it declares $b_j = v_j$ is equal to:

$$u(b_j) = \sum_{i\in\mathcal{M}}\sum_{n\in\mathcal{A}} x_{i,n}^{-j} d_i \left[C\left(1-h_n\right) - P\right] + \sum_{\substack{n\in\mathcal{A}\\ n\neq j}} y_n^{-j} b_n + \\ + \sum_{i\in\mathcal{M}}\sum_{n\in\mathcal{A}} x_{i,n} d_i \left[P - C\left(1-h_n\right)\right] - \sum_{n\in\mathcal{A}} y_n b_n, \tag{3.19}$$

While, when it declares $\hat{b}_j \neq v_j$ it has a utility equal to:

$$u(\hat{b}_j) = \sum_{i\in\mathcal{M}}\sum_{n\in\mathcal{A}} x_{i,n}^{-j} d_i \left[C\left(1-h_n\right) - P\right] + \sum_{\substack{n\in\mathcal{A}\\ n\neq j}} y_n^{-j} b_n + \\ + \sum_{i\in\mathcal{M}}\sum_{n\in\mathcal{A}} \hat{x}_{i,n} d_i \left[P - C\left(1-h_n\right)\right] - \sum_{n\in\mathcal{A}} \hat{y}_n b_n + \hat{y}_n v_j - v_j. \tag{3.20}$$

Given the objective function of MCAP, (x, y) is the solution that maximizes (3.4), that is $(x, y) = \operatorname{argmax} \sum_{i\in\mathcal{M}}\sum_{n\in\mathcal{A}} x_{i,n} d_i \left[P - C\left(1-h_n\right)\right] - \sum_{n\in\mathcal{A}} y_n b_n$. Moreover $v_j - \hat{y}_n v_j \geq 0$, in fact $\hat{y}_n \in \{0, 1\}$. Therefore, since:

$$u(b_j) - u(\hat{b}_j) = \sum_{i\in\mathcal{M}}\sum_{n\in\mathcal{A}} x_{i,n} d_i \left[P - C\left(1-h_n\right)\right] - \sum_{n\in\mathcal{A}} y_n b_n - \\ - \left\{\sum_{i\in\mathcal{M}}\sum_{n\in\mathcal{A}} \hat{x}_{i,n} d_i \left[P - C\left(1-h_n\right)\right] - \sum_{n\in\mathcal{A}} \hat{y}_n b_n\right\} + v_j - \hat{y}_n v_j, \tag{3.21}$$

we have that $u(b_j) - u(\hat{b}_j) \geq 0$.

□

Discussion

Since our proposed formulation fosters the partial coverage of nodes, the value of the profit parameter P has a remarkable impact on the solution of the allocation scheme. As a matter of fact, by setting P very small, the optimal solution for the CP will be to serve no clients at all. On the other hand, if P is set large enough, the solution will serve all the clients' demands, given the available capacity of the networking infrastructure. The following proposition establishes exactly this trade-off.

Proposition 1. *Using the objective function (3.4), the model associates the highest possible number of MCs if P satisfies:*

$$P > \max_{j\in\mathcal{A}} b_j + C \max_{i\in\mathcal{M}} d_i. \tag{3.22}$$

Proof. We show that the proposition holds, by considering the worst case: a scenario where the most expensive AP is selected, even though it does not provide any storage.

Algorithm 2: Greedy Cache and Bandwidth Auction

Input : $\mathcal{M}, \mathcal{A}, \mathbf{b}, \mathbf{s}, \mathbf{d}, \mathbf{r}, \mathbf{h}, \mathbf{R}, C, P$
Output: $\mathbf{y}, \mathbf{p}, \mathbf{x}$
1 $(\mathbf{y}, \mathbf{x}, c) \Leftarrow$ Greedy_Allocation($\mathcal{M}, \mathcal{A}, b_j, s_j, d_i, r_{ij}, h_j, R_j, C, P$);
foreach $j \in \mathcal{A} : y_j = 1$ **do**
2 $\quad p_j \Leftarrow \frac{b_c}{|\mathcal{M}_c|}|\mathcal{M}_j|$;
end

Let $j \in \mathcal{A}$ be the access point that made the highest possible bid, and let $h_j = 0$ be the cache hit-rate that we get at AP j. Let $i \in \mathcal{M}$ be the mobile client that generates the highest traffic demand among all the others (i.e., $\underset{i \in \mathcal{M}}{\operatorname{argmax}}\, d_i$). In the most expensive case, the CP activates j only to serve the demand of i incurring a cost $C_{max} = \max_{j \in \mathcal{A}} b_j + C \max_{i \in \mathcal{M}} d_i$. Thus, by setting $P > C_{max}$ we ensure that if the network has enough spare capacity, the optimal allocation rule will serve this request, since this choice will certainly lead to an improvement in the value of the objective function. □

3.4 Greedy Algorithms

The computational complexity of the MCAP problem is such that, as the number of MCs $|\mathcal{M}|$ and APs $|\mathcal{A}|$ increases, the completion time of the optimal allocation (i.e., Algorithm 1) raises exponentially. In order to find an allocation strategy that can effectively scale up to large network instances, this section provides a computationally efficient, polynomial-time algorithm with three alternative strategies, that still guarantees *truthfulness* and *individual rationality*. We begin by describing the *greedy MC* alternative, while we will talk about the other greedy strategies at the end of the section.

Let $k_j = \frac{b_j}{|\mathcal{M}_j|}$ be the ratio between the bid of the j-th AP (b_j) and the number of MCs that it can potentially serve, $|\mathcal{M}_j|$, given the radio access coverage constraints only. The quantity k_j can be interpreted as the price per mobile client that should be paid to the j-th AP if it served all the MCs in its range. Let c be the index of the *critical access point*, which is defined as the first AP that has not been selected by the mechanism. The input values shared by the algorithms (i.e., $\mathcal{M}, \mathcal{A}, \mathbf{b}, \mathbf{s}, \mathbf{d}, \mathbf{r}, \mathbf{h}, \mathbf{R}, C, P$), are the same parameters we used to characterize the optimal allocation problem as in Table 3.1.

In the same way as in the optimal auction, the greedy algorithm (*Algorithm 2*) is characterized by the following two phases: 1. the *allocation* phase, which selects the APs characterized by the lowest $k_j = \frac{b_j}{|\mathcal{M}_j|}$ until the demands of MCs are satisfied (Step 1, Alg. 2) , and 2. the *payment* phase, which determines the price paid to the selected APs, given the characteristics of the critical access point c (Step 2, Alg. 2).

The greedy allocation of MCs to APs, as well as the identification of the critical access point c, are performed by *Algorithm 3*. The allocation procedure (Alg. 3) starts by removing the non-profitable APs (Step 1, Alg. 3), and sorts the APs in non-decreasing order according to the corresponding k_j values (Step 2, Alg. 3). This is done with the

Algorithm 3: Greedy Allocation (Step 1 of Alg. 2)

Input : $\mathcal{M}, \mathcal{A}, \mathbf{b}, \mathbf{s}, \mathbf{d}, \mathbf{r}, \mathbf{h}, \mathbf{R}, C, P$
Output: $\mathbf{y}, \mathbf{x}, c$
1 $\mathcal{A} \Leftarrow \{j \in \mathcal{A} : P - C\,(1 - h_j) \geq 0\}$;
2 $L \Leftarrow Sort\left(j \in \mathcal{A}, \frac{b_j}{|\mathcal{M}_j|}, \text{“non-decreasing”}\right)$;
while $\exists i \in \mathcal{M} : \sum_{j \in \mathcal{A}} x_{i,j} < 1$ **do**
  $j \Leftarrow Next(L)$; $y_j \Leftarrow 1$;
3 $\mathcal{V}_j \Leftarrow$ Sort $\left(k \in \mathcal{M}_j, \frac{d_k}{r_{kj}}, \text{“non-decreasing”}\right)$;
  foreach $k \in \mathcal{V}_j$ **do**
    $x_{kj} \Leftarrow 1$;
4 **if** $\neg$*Is_Feasible_Solution*$(\mathcal{M}, \mathcal{A}, \mathbf{x}, \mathbf{y}, \mathbf{b}, \mathbf{s}, \mathbf{d}, \mathbf{r}, \mathbf{h}, \mathbf{R}, j)$ **then** $x_{kj} \Leftarrow 0$;
  end
end
5 $c \Leftarrow Next(L)$;
6 **foreach** $j \in Reverse(L)$ **do**
  if $\sum_{i \in \mathcal{M}} x_{ij} = 0$ **then** $y_j \Leftarrow 0$; $c \Leftarrow j$;
  else break;
end

aim of selecting the *“most promising”* APs in terms of the declared price per number of reachable mobile clients. We then iteratively allocate to the APs the largest number of unassigned mobile clients by choosing those which have lower capacity demand $\frac{d_i}{r_{ij}}$ (Step 3, Alg. 3), while still preserving the feasibility of the solution (Step 4, Alg. 3). Before completing the execution of Alg. 3, in Steps 5 and 6 we set the value of the critical access point, which is the first AP that has not been allocated any MC by the algorithm (the first looser).

Finally, *Algorithm 4* checks if the solution is feasible, by determining whether the constraints of the ILP model hold. In detail, Alg. 4 checks in Step 1 if each MC is assigned to at most one AP, as imposed by constraints (3.5); Step 2 checks whether the available Wi-Fi bandwidth is not saturated, as in constraints (3.6), and lastly Step 3 checks if the backhaul connection can accommodate the allocated traffic, as in constraints (3.7).

Alg. 2 implements a *truthful* auction since 1. the allocation phase respects the *monotonicity property* as the APs are sorted in non-increasing order of their bids per number of covered mobile customers, and 2. there exists a *critical value* which determines whether the AP has been selected or not. As demonstrated in [137] the previous two conditions ensure the truthfulness of the greedy algorithm.

By jointly substituting Step 2 of Alg. 2 and Step 2 of Alg. 3, as detailed in Table 3.2, we can change the optimization strategy of the greedy algorithm; in particular, we propose the following three variants:

1. *Greedy MC* (G.m.);

Algorithm 4: Is Feasible Solution (Step 4 of Alg. 3)

Input : $\mathcal{M}, \mathcal{A}, \mathbf{x}, \mathbf{y}, \mathbf{b}, \mathbf{s}, \mathbf{d}, \mathbf{r}, \mathbf{h}, \mathbf{R}, j$
Output: c
$c_1 \Leftarrow \sum_{\forall j \in A} x_{i,j} \leq 1$;
$c_2 \Leftarrow \sum_{\forall i \in \mathcal{M}} \frac{d_i x_{i,j}}{r_{i,j}} \leq 1$;
$c_3 \Leftarrow \sum_{\forall i \in \mathcal{M}} d_i x_{i,j}(1 - h_j) \leq R_j$;
$c \Leftarrow c_1 \wedge c_2 \wedge c_3$;

2. *Greedy cache* (G.c.);
3. *Greedy backhaul* (G.b.).

As discussed previously, the *greedy MC* (G.m.) alternative gives higher priority to the APs that can potentially serve more MCs, due to their space location. On the other hand, the *greedy cache* (G.c.) selects the APs that are leasing larger caching storage; lastly, the *greedy backhaul* (G.b.) chooses the APs with a faster backhaul Internet connection. Table 3.2 summarizes these three variants, which will be compared numerically in Sec. 3.6. Finally, we observe that all the variants we propose do not modify the monotonicity property of the allocation phase; therefore, it is straightforward to show that *truthfulness* and *individual rationality* still hold.

3.5 Network Selection Game

Our proposed optimal allocation strategy and the corresponding heuristic algorithms are formulated for a *centralized* setting in which the content provider (CP) optimally allocates the mobile clients (MCs) to the access points (APs). However, in a real scenario, MC owners typically choose autonomously which access point should their device connect to. At the same time, each user's choice will influence the others: as a matter of fact, by changing the AP at which a given MC is connected, the MC owner is indeed affecting the quality of the wireless channel perceived by the other users. In this scenario, game theory is the natural tool used to model the players' mutual interaction (e.g., [64, 139]). In particular, in this section we consider a *fully distributed case*, and we model this practical scenario as a *non-cooperative network selection game* in which the players (i.e., the MCs) interact with each other by selfishly selecting the access

Table 3.2: *Summary of the 3 proposed variants for the greedy algorithm*

Greedy Variant	Step 2, Alg. 2	Step 2, Alg. 3
MC (G.m.)	$p_j \Leftarrow \frac{b_c}{\lvert\mathcal{M}_c\rvert}\lvert\mathcal{M}_j\rvert$	$L \Leftarrow \text{Sort}\left(j \in \mathcal{A}, \frac{b_j}{\lvert\mathcal{M}_j\rvert}, \text{“non-decr”}\right)$
Cache (G.c.)	$p_j \Leftarrow \frac{b_c}{h_c} h_j$	$L \Leftarrow \text{Sort}\left(j \in \mathcal{A}, \frac{b_j}{h_j}, \text{“non-decr”}\right)$
Backhaul (G.b.)	$p_j \Leftarrow \frac{b_c}{R_c} R_j$	$L \Leftarrow \text{Sort}\left(j \in \mathcal{A}, \frac{b_j}{R_j}, \text{“non-decr”}\right)$

network (i.e., the AP) that minimizes their cost. We study this scenario to derive the performance loss that the non-cooperative, distributed allocation causes, compared to the centralized global optimum.

In this scenario, a mobile client $i \in \mathcal{M}$ can be served by multiple APs in its current location. Different APs will likely provide different utilities to the players, and, at the same time, the utility will be a function of the other players' choices. In such distributed setting, all the MCs that can be served by at least one AP will connect to the wireless infrastructure. On the other hand, all the mobile clients that are not covered by the signal of an activated access point will not participate to the network selection game.

For the sake of simplicity, hereafter we denote with $\mathcal{A}$ the set of activated access points, and with $\mathcal{M}$ the set of mobile clients that can be reached by at least one AP. The quality metric that we consider relevant for the MCs is the *congestion level* experienced while connecting to a specific AP. In particular, due to the well known 802.11 anomaly [91], we assume that the rate experienced by all the MCs connected to an AP $j \in \mathcal{A}$ is the same and it is equal to $r_j = \min_{i \in \mathcal{M} | l_{i,j} \leq L_j} r_{i,j}$, that is the rate of the farthest MC potentially served by the AP. We therefore define the cost function of player $i \in \mathcal{M}$ as follows: $c_i(x) = \sum_{\forall k \in M, \forall j \in A} \frac{x_{i,j} \cdot x_{k,j}}{r_j}$.

Let $S = \mathbf{x}$ be a strategy profile. The choice of player $i \in \mathcal{M}$ is represented with $\mathbf{x}_i$, while the one of the other players is $\mathbf{x}_{-i}$. In particular, the binary variable $x_{i,j} = 1$ if and only if MC i is connected to AP j. The solution concept we adopt is the pure strategy Nash Equilibrium (NE): a strategy profile S in which there is no player that can improve its utility by unilaterally deviating from his choice.

In this section we prove that the network selection game is weighted potential, and therefore by letting players play the best response dynamics, one Nash equilibrium will be reached. Furthermore, we also prove that there exists only one such Nash Equilibrium.

The rationale behind the following proofs is as follows: consider the mobile client $i \in \mathcal{M}$, and assume that it is currently connected to the AP $j \in \mathcal{A}$. The cost (i.e., the congestion) perceived by such mobile client is $c_i(x) = \sum_{\forall k \in \mathcal{M}, \forall j \in \mathcal{A}} \frac{x_{i,j} \cdot x_{k,j}}{r_j} = \sum_{i \in \mathcal{M}} \frac{x_{i,j}}{r_j}$. A Nash Equilibrium is a strategy profile such that if MC i migrates to another AP $k \neq j$, it should observe a higher cost (i.e.: $\frac{1}{r_k} + \sum_{i \in \mathcal{M}} \frac{x_{i,k}}{r_k} > \sum_{i \in \mathcal{M}} \frac{x_{i,j}}{r_j}, \forall k \in \mathcal{M}, k \neq i$).

Hereafter we show that the network selection game is weighted potential, and therefore the distributed solution obtained with the *best response dynamics* converges to a NE [134].

Proposition 2. ***Weighted Potential:*** *Our proposed network selection game is weighted potential with potential function* $\Phi(\mathbf{x}) = \sum_{k \in \mathcal{M}} \sum_{j \in \mathcal{A}} \sum_{i \in \mathcal{M}} \frac{x_{k,j} x_{i,j}}{r_j} + \sum_{j \in \mathcal{A}} \frac{1}{r_j} \sum_{i \in \mathcal{M}} x_{i,j}$.

Proof. To prove that our proposed game is weighted potential, we must show that, given the strategy space $\mathcal{S}$ there is a potential function $\Phi : \mathcal{S} \rightarrow \mathbb{R}$ and a vector $\mathbf{w} \in \mathbb{R}_+^{|\mathcal{M}|}$ such that $\forall \mathbf{x}_{-i} \in \mathcal{S}_{-i}, \forall \mathbf{x}_i, \mathbf{x}'_i \in \mathcal{S}_i$ it holds that

$$\Phi(\mathbf{x}_i, \mathbf{x}_{-i}) - \Phi(\mathbf{x}'_i, \mathbf{x}_{-i}) = w_i(u_i(\mathbf{x}_i, \mathbf{x}_{-i}) - u_i(\mathbf{x}'_i, \mathbf{x}_{-i})) \tag{3.23}$$

Consider the AP $i \in \mathcal{A}$. Let $x_{i,j} = \begin{cases} 1 & \text{if } j = h \\ 0 & \text{otherwise} \end{cases}$, while $x'_{i,j} = \begin{cases} 1 & \text{if } j = q \\ 0 & \text{otherwise} \end{cases}$. In other words, $\mathbf{x}_i$ states that MC i is connected to AP h, whereas $\mathbf{x}'_i$ states that MC i is connected to AP q. Let $n_{i,j} = \sum_{w \in \mathcal{M} \setminus \{i\}} x_{w,j}$ be the number of MCs, excluding i, connected to AP j. Then we have:

$$u_i(\mathbf{x}_i, \mathbf{x}_{-i}) - u_i(\mathbf{x}'_i, \mathbf{x}_{-i}) = \frac{r_q + r_q n_{i,h} - r_h - r_h n_{i,q}}{r_h r_q}. \tag{3.24}$$

Moreover, considering the potential function we obtain:

$$\begin{aligned} \Phi(\mathbf{x}_i, \mathbf{x}_{-i}) - \Phi(\mathbf{x}'_i, \mathbf{x}_{-i}) &= \frac{(n_{i,h}+1)^2 + (n_{i,h}+1)}{r_h} + \frac{(n_{i,q})^2 + n_{i,q}}{r_q} \\ &- \frac{(n_{i,h})^2 + n_{i,h}}{r_h} - \frac{(n_{i,q}+1)^2 + (n_{i,q}+1)}{r_q} = \\ &= 2\left(\frac{r_q + r_q n_{i,h} - r_h - r_h n_{i,q}}{r_h r_q}\right), \end{aligned} \tag{3.25}$$

and therefore the game is *weighted potential* since it is sufficient to set $\forall i \in \mathcal{A}, w_i = 2$. □

The existence of a Nash Equilibrium is guaranteed by the previous Proposition; we now formally prove that there exists only one Nash Equilibrium. In particular, the following Theorem proves that in our proposed network selection game, Nash Equilibria are "*essentially unique*" [137], meaning that all equilibrium configurations have the same overall cost.

Theorem 4. ***Uniqueness of the Nash Equilibrium:*** *Our proposed network selection game has an essentially unique Nash Equilibrium.*

Proof. Let $n_j = \sum_{k \in \mathcal{M}} x_{i,j}$ be the number of users simultaneously connected to the AP $j \in \mathcal{A}$; for the sake of clarity, we can then rewrite the potential function as follows:

$$\Phi(\mathbf{x}) = \sum_{k \in \mathcal{M}} \sum_{j \in \mathcal{A}} \sum_{i \in \mathcal{M}} \frac{x_{k,j} x_{i,j}}{r_j} + \sum_{j \in \mathcal{A}} \frac{1}{r_j} \sum_{i \in \mathcal{M}} x_{i,j} = \sum_{j \in \mathcal{A}} \frac{n_j^2}{r_j} + \sum_{j \in \mathcal{A}} \frac{n_j}{r_j} = \Phi(\mathbf{n}). \tag{3.26}$$

By the definition of the potential function itself [137], Nash Equilibria can be computed by solving the optimization problem: $\min \Phi(\mathbf{n})$, subject to $\sum_{j \in \mathcal{A}} n_j = |\mathcal{A}|$. One such optimization problem is a convex Mixed-Integer Non-Linear Program, in fact, the continuous relaxation of the objective function is convex.

For this class of problems, a local optimal solution is also equal to the global optimum [26, 33], thus the Nash Equilibrium is essentially unique.

□

The Price of Anarchy (PoA) is the ratio between the worst Nash Equilibrium and the centralized optimal solution. On the other hand, the Price of Stability (PoS) is the ratio between the best NE and the centralized optimal solution. Since we proved uniqueness of Nash Equilibrium, in our proposed network selection game $PoA = PoS$. Hereafter we show that a lower bound for the PoA is $\frac{4}{3}$.

Proposition 3. ***Lower bound on the PoA.*** *The Price of Anarchy in our proposed network selection game is* $PoA \geq \frac{4}{3}$.

Proof. Consider a network scenario with 2 access points and 2 mobile clients, and assume that the first AP has a rate of $r_1 = 1$, while the second has a rate of $r_2 = 2 + \epsilon$, for a small $\epsilon > 0$.

The centralized optimal solution is obtained when each MC connects to one of the available APs and, in this scenario, the overall congestion is $\text{Cong}_{OPT} = 1 + \frac{1}{2+\epsilon}$.

On the other hand, the Nash Equilibrium is obtained when both the MCs connect to the AP with the higher rate. As a matter of fact, in this configuration, each MC experiences a congestion equal to $\frac{2}{2+\epsilon} < 1$, where 1 is the congestion the MC would experience if it migrated to the other access point. In the NE configuration, the overall congestion is given by $\text{Cong}_{NE} = 2^2/(2+\epsilon)$.

Therefore, since the Price of Anarchy is $PoA = \frac{4/(2+\epsilon)}{1+\frac{1}{2+\epsilon}}$, a lower bound for it is $PoA \geq 4/3$, obtained when $\epsilon \to 0^+$.

□

To provide further evidence that the previous bound is very tight, we numerically computed values for the PoA. In order to do that, we formulated a Mixed-Integer Non-Linear Optimization model (omitted here for the sake of brevity) and we used the Bonmin solver [26] to numerically find solutions to one such optimization problem. We numerically confirm that the Price of Anarchy is $PoA = 4/3$ (i.e., the bound we provided for the PoA is tight), for each number of access points up to 50.

3.6 Numerical Results

In this section, we analyze and discuss the numerical results obtained solving our proposed models and heuristics in realistic, large-size network scenarios. More specifically, we first illustrate (Sec. 3.6.1) the experimental methodology used to evaluate the proposed algorithms, and then (Sec. 3.6.2) we discuss the obtained results.

3.6.1 Methodology

Unless stated otherwise, simulations are conducted by uniformly distributing 60 access points (APs) in a 300×300 m^2 grid, while the locations of 100 mobile clients (MCs) are selected as a bi-variate Gaussian distribution centered on uniformly chosen APs. The bids of the APs are selected uniformly in the $[7, 15]$ USD range, whereas the demands of every MC are generated in the range $[0.5, 3]$ Mbit/s. We set the traffic-proportional cache miss cost C to 5 USD per Mbit/s, and we consider different profit

values P, in the set $\{5, 10, 15, 20, 25\}$ USD per Mbit/s. We performed 20 runs for every scenario, computing the narrow 95% confidence intervals reported in each figure.

We assume that the radio access network is composed of 802.11a wireless access points. We consider the effects of path attenuation, where the distance between the mobile client (MC) $i \in \mathcal{M}$ and an access point (AP) $j \in \mathcal{A}$ is $l_{i,j}$, with a frequency of 5.25 GHz, a path loss exponent of 3 and the radio specification of the Atheros AR5413 chipset. In particular we can compute the transmission rate by comparing the path loss value to the receiver sensitivity provided in Table 3.3.

The Zipf discrete distribution is used to characterize the object popularity, since it was shown that it is an adequate model for it [43, 79, 178]. In particular, the Zipf model partitions the objects in N groups according to their popularity rank. Let α be the popularity exponent of the distribution, the Zipf probability mass function (PMF) and cumulative density function (CDF) for the n-th popularity rank are defined as:

$$P(X = n) = \frac{1/n^{\alpha}}{\sum_{i=1}^{N} (1/i^{\alpha})}, \tag{3.27}$$

$$F(n) = \sum_{i=1}^{n} P(n) = \frac{\sum_{i=1}^{n} (1/i^{\alpha})}{\sum_{i=1}^{N} (1/i^{\alpha})}. \tag{3.28}$$

As frequently done in the literature (e.g: [38, 77, 153]) we work under the assumptions of the *"Independent Reference Model"* (IRM), according to which successive content requests are independent identically distributed (i.i.d.), where the popularity class of an object is a discrete random variable X, with probability mass function $P(X)$. We assume that cache replacement is performed according to the *"Least frequently used"* (LFU) policy, since under the IRM assumption, LFU is known to maximize the hit rate [77].

Let C_j be the rank of the least popular cached objects at the j-th AP. Since popularity classes are sorted in decreasing order, C_j can be computed as follows:

$$C_j = \left\lfloor \frac{\text{Cache size at AP } j}{\text{Average object size}} \right\rfloor. \tag{3.29}$$

Let $q(n)$ be the request rate of the objects belonging to the n-th popularity class. We can then define the hit rate h_j for AP j as:

$$h_j = \frac{\sum_{1 \leq n \leq C_j} q(n)}{\sum_{1 \leq n \leq N} q(n)}, \tag{3.30}$$

which is the ratio between the requests for objects belonging to the C_j most popular classes that have been stored in the caches and the requests for all the available objects.

Table 3.3: *Receiver sensitivity of the Atheros AR5413 chipset*

Path Loss (dB)	-72	-73	-77	-81	-84	-86	-88	-90
Rate (Mbit/s)	54	48	36	24	18	12	9	6

Finally, we define the *normalized cache size*, N, as the ratio between the average cache size and the size of all the objects:

$$N = \frac{\text{Avg. cache size}}{\text{Objects cardinality} \cdot \text{Average object size}}. \tag{3.31}$$

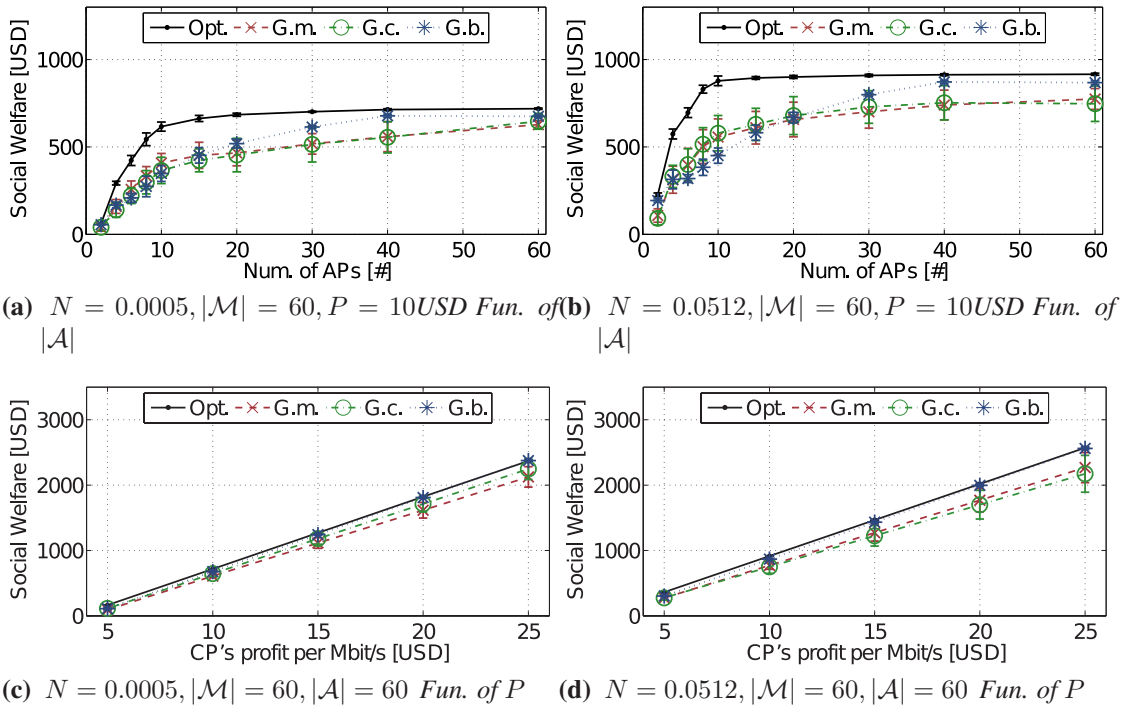
For each AP, we chose the backhaul Internet bandwidth uniformly in the set $\{1; 6; 8; 20; 100\}$ Mbit/s, while the size of the leased caching storage was uniformly selected in the range $[10, 100]$ Gbytes. We consider objects having an average size of 100 Mbytes, while we take into account two catalog cardinalities equal to 10^4 and 10^6 objects, in line with what Fricker et al. have identified for the User Generated Content (UGC) scenario in [77]. However, for similar values of N, similar trends can be observed either by changing the object catalog size, or by increasing the amount of caching storage that is leased.

In the following experiments we set the Zipf α exponent to 0.9, in line with [77]. Even though better results can be observed with higher α values, due to the higher skewness of the request distribution, we cautiously consider this more adverse case. Since we assume that the object cardinality can either be 10^4 or 10^6, the normalized cache sizes N are thus equal to 0.0512 and 0.0005 respectively, meaning that caches can store up to 5.12% of the entire catalog, when 10^4 objects are considered, while only 0.05% of the objects can be cached, while considering a larger catalog of 10^6 objects. Results for $N = 0.0051$ were also generated, but for the sake of brevity will be omitted here since they are in between the two extremes. Regarding the three variants of the algorithms presented in Table 3.2, we denote the greedy *MC*, *cache* and *backhaul* with G.m., G.c. and G.b., respectively.

3.6.2 Performance Analysis

Unless otherwise specified, the figures illustrated in this section are related to a heterogeneous network with 60 mobile clients (MCs) and 60 access points (APs), with a profit $P = 10$USD per Mbit/s.

Social Welfare. The *social welfare* metric (SW) is the objective function of the ILP model (3.4), and has a monetary currency (USD) as unit of measurement. The value of such metric is portrayed in Fig. 3.2. In particular, Fig. 3.2a and 3.2b show the SW trend as a function of the number of APs, for a content catalog of 10^6 ($N = 0.0005$) and 10^4 objects ($N = 0.0512$), respectively. On the other hand Fig. 3.2c and 3.2d show the SW as a function of the CP's profit per unit of bandwidth. The SW has an increasing trend with respect to the number of access points $|\mathcal{A}|$, as well as the average profit of the content provider P; in fact, when the number of APs, or the profit increase, the allocation algorithm chooses to accommodate more traffic demands since it improves the value of the SW metric. We also note that, while the SW has a linear trend in the profit (as in Fig. 3.2c-3.2d), it exhibits instead a logarithmic trend in the number of APs (as in Fig. 3.2a-3.2b). This behavior is caused by the fact that having $|\mathcal{A}| < 10$ significantly penalizes the performance of the system because the network infrastructure does not have enough capacity to serve all the clients' demands.



(a) $N = 0.0005, |\mathcal{M}| = 60, P = 10$*USD Fun. of* $|\mathcal{A}|$ **(b)** $N = 0.0512, |\mathcal{M}| = 60, P = 10$*USD Fun. of* $|\mathcal{A}|$

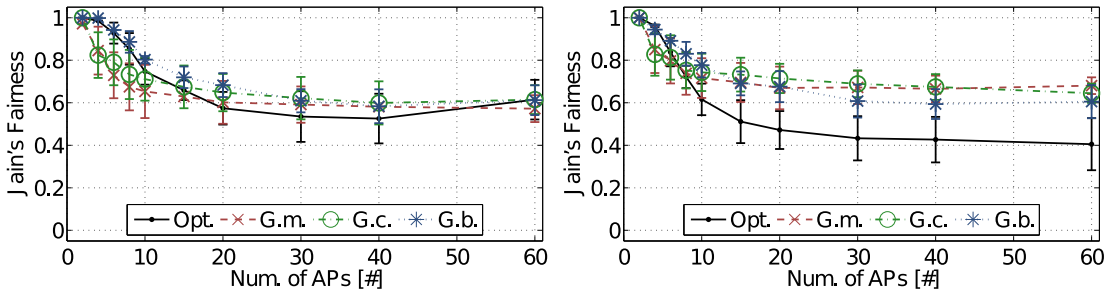
(c) $N = 0.0005, |\mathcal{M}| = 60, |\mathcal{A}| = 60$ *Fun. of* P **(d)** $N = 0.0512, |\mathcal{M}| = 60, |\mathcal{A}| = 60$ *Fun. of* P

Figure 3.2: Social Welfare*, The plots show the value obtained for the Social Welfare* (SW) *metric, comparing the value obtained with the optimal allocation algorithm and the three greedy heuristics. In particular, in Fig. 3.2a and 3.2b we show the trend of the* SW *as a function of the number of APs, for a catalog of* 10^6 $(N = 0.0005)$ *and* 10^4 $(N = 0.0512)$ *objects, respectively. Fig. 3.2c and 3.2d show the effect of the profit.*

Another interesting observation that can be deduced by comparing Fig. 3.2a and 3.2b is that having a larger normalized cache size of $N = 0.0521$ (10^4 objects) can lead to significant improvements on the SW which is 27% higher than for $N = 0.0005$ (10^6 objects), when considering 60 APs. Moreover, considering Fig. 3.2c and 3.2d, we can conclude that the higher the profit per unit of bandwidth, the lower the gain of having a larger N. In particular, in this latter case, when the profit is 25USD per Mbit/s, the gain for $N = 0.0512$ is only 9% with respect to $N = 0.0005$.

Finally, comparing the values obtained for the optimal allocation algorithm and our proposed heuristics, we observe in Fig. 3.2c-3.2d that, when considering the SW as a function of the profit, the heuristics compute a close-to-optimal value for the objective function, losing, in the worst case, only 16% of SW. In particular, the *greedy backhaul* (G.b.) allocation algorithm seems to perform slightly better than the other two, efficiently computing a solution on average less than 10% lower than the optimum. Similar conclusions regarding the greedy heuristics can be derived from Fig. 3.2a-3.2d.

Jain's Fairness Index. The Jain's Fairness Index (f) is defined according to [101]



(a) $N = 0.0005, |\mathcal{M}| = 60, P = 10USD$ - f **(b)** $N = 0.0512, |\mathcal{M}| = 60, P = 10USD$ - f

Figure 3.3: Jain's Fairness Index, *Fig. 3.3a shows Jain's Fairness Index f for $N = 0.0005$, as a function of the number of APs, whereas in Fig. 3.3b we portrait the result for $N = 0.0512$.*

as:

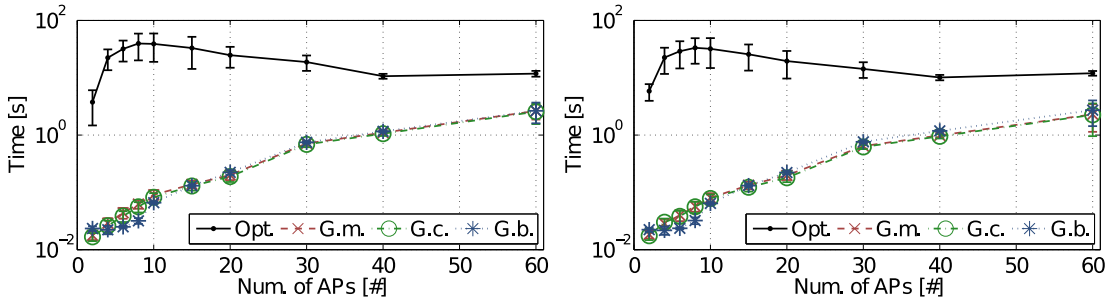
$$f = \frac{\left(\sum_{\forall j \in \mathcal{A}} \rho_j\right)^2}{\left(\sum_{\forall j \in \mathcal{A}} y_j\right) \cdot \sum_{\forall j \in \mathcal{A}} \rho_j^2}$$

where $\rho_j = \frac{y_j p_j}{\sum_{\forall i \in \mathcal{M}} x_{i,j} d_i}$ is the ratio between the price paid to lease AP j and the overall demand it is serving. The performance of the optimal and greedy allocation algorithms with respect to the f metric is depicted in Fig. 3.3, where in 3.3a we set the normalized cache size $N = 0.0005$ (10^6 objects), whereas in 3.3b, $N = 0.0512$ (10^4 objects).

By comparing the two figures, we can state that the greedy heuristics have a similar trend with respect to the fairness metric, which is rather insensitive to the normalized cache size. As a matter of fact, in both Fig. 3.3a and 3.3b we observe that the heuristics generate a solution that is on average 62% fair, when the number of AP is $|\mathcal{A}| = 60$. At the same time, the optimal allocation algorithm leads to worse solutions in terms of the fairness metric, especially when having a larger cache. In fact, in this latter case, as shown in Fig. 3.3b, the fairness of the optimal allocation algorithm stabilizes at 40% when 60 APs are considered.

Completion Time. The effect of the completion time (t) is shown in Fig. 3.4. As usual, Fig. 3.4a refers to $N = 0.0005$ (10^6 objects) while Fig. 3.4b is for $N = 0.0512$ (10^4 objects). First of all, by comparing Fig. 3.4a with 3.4b, we can conclude that the size of the catalog, and therefore the average AP's hit-rate has a negligible impact on the observed completion time. On top of that, the optimal allocation algorithm requires much more time than the one spent using our proposed heuristics, which are all characterized by practically the same computing time.

The time trend of the optimal allocation algorithm not only has a local maximum when the number of access points $|\mathcal{A}|$ is equal to 9, but it also exhibits large confidence intervals when $|\mathcal{A}| < 10$. The reason for both these behaviors lies in the fact that when there are few access points (APs), the allocation will not serve all the clients' demands,



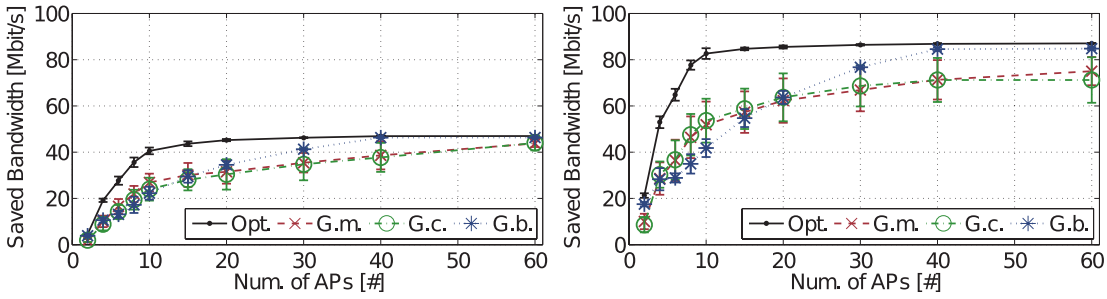
(a) $N = 0.0005, |\mathcal{M}| = 60, P = 10$*USD* - t **(b)** $N = 0.0512, |\mathcal{M}| = 60, P = 10$*USD* - t

Figure 3.4: Completion Time*, The completion time t metric is shown as a function of the number of APs. In Fig. 3.4a we report the results for $N = 0.0005$, whereas Fig. 3.4b refers to $N = 0.0512$.*

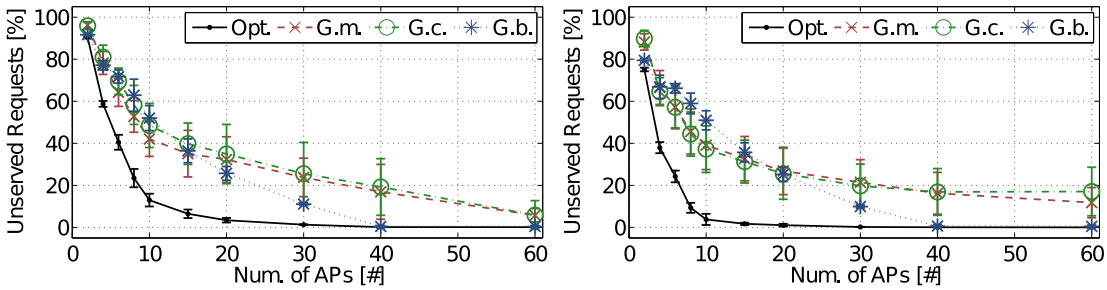
since the spare network capacity is not sufficient to do so. Moreover, the choice of which AP should be used becomes critical, and more time is required by the algorithm in order to explore the combinatorial state space of the solutions.

Saved Bandwidth, Unserved Requests, Hit Rate. Hereafter we jointly discuss the results we obtained for the SB, UR and $\bar{h}$ metrics. In particular, the Bandwidth Saved for Caching (SB) is defined as the total amount of traffic directly served by the caches, measured in Mbit/s and equal to $SB = \sum_{\substack{\forall i \in \mathcal{M} \\ \forall j \in \mathcal{A}}} x_{i,j} d_i h_j$. This metric provides further insights on the cost and energy savings that the operator can experience by reducing the load on his distribution infrastructure. The UR metric is instead the Fraction of Unserved Requests, which is the fraction of the mobile clients' traffic demands that are not accommodated according to the allocation algorithm, and is equal to $UR = \frac{\sum_{\forall i \in \mathcal{M}} \left(1 - \max_{\forall j \in \mathcal{A}} x_{i,j}\right) d_i}{\sum_{\forall i \in \mathcal{M}} d_i}$. Finally, the Average Hit Rate $\bar{h}$ is defined as the ratio between the content served by the caches and the sum of all the served clients' demands, and is equal to $\bar{h} = \frac{SB}{\sum_{\substack{\forall i \in \mathcal{M} \\ \forall j \in \mathcal{A}}} x_{i,j} d_i}$.

The Saved Bandwidth (SB) trend, shown in Fig. 3.5a and 3.5b is logarithmic in the number of APs deployed. However, the average hit rate $\bar{h}$ observed is constant in the number of access points and equal to 41% with a normalized cache size of $N = 0.0005$ and 78% when $N = 0.0521$, for both the optimal allocation algorithm as well as the heuristics. The constant value for $\bar{h}$ and the logarithmic trend for SB are caused by the fact that our proposed formulation admits partial coverage, that is, not all the requests of the mobile clients (MCs) are served by the network. In particular, as Fig. 3.5c and 3.5d show, when $N = 0.0005$, 10 APs can serve up to 84% of the overall demand (using the optimal algorithm), while 96% is reached when $N = 0.0521$. Since the bottleneck in our system model is the backhaul internet connection of every access point, deploying some caching storage in the system makes the network accommodate more traffic. On top of that, 40 APs are necessary to make all the client's requests be served with $N =$



(a) $N = 0.0005, |\mathcal{M}| = 60, P = 10 USD$ - SB **(b)** $N = 0.0512, |\mathcal{M}| = 60, P = 10 USD$ - SB



(c) $N = 0.0005, |\mathcal{M}| = 60, P = 10 USD$ - UR **(d)** $N = 0.0512, |\mathcal{M}| = 60, P = 10 USD$ - UR

Figure 3.5: Saved Bandwidth, Unserved Requests*, Fig. 3.5a and 3.5b show the Saved Bandwidth metric as a function of the number of APs, for* $N = 0.0005$ *and* $N = 0.0521$*, respectively. The trend of the fraction of Unserved Requests is instead depicted in Fig. 3.5c and 3.5d.*

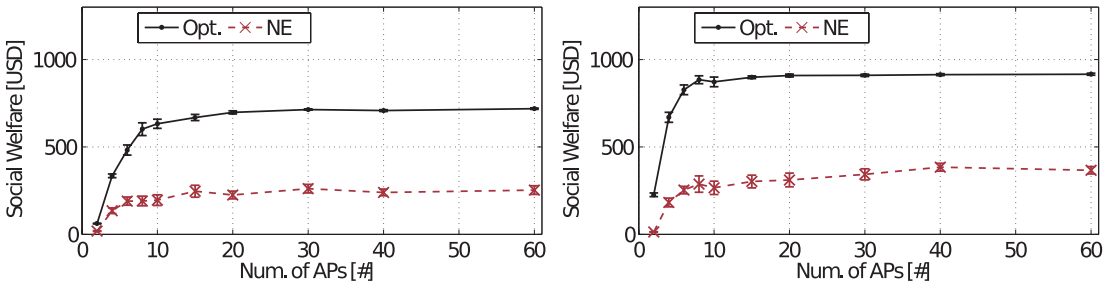
0.0005 (10^6 objects), whereas 30 APs are sufficient if we consider $N = 0.0521$ (10^4 objects). In terms of the unserved requests, the greedy backhaul very well approaches the results observed for the optimal solution, whereas the other two heuristic algorithms serve on average 18% less traffic demands, when $N = 0.0521$ and $|\mathcal{A}| = 60$.

As we have already commented for the SW, LC and t metric, the threshold of 10 APs clearly marks two different operating zones: when a large amount of traffic requests cannot be served by the networking infrastructure, this has a major impact on the multiple metrics characterizing the overall system performance.

Social Welfare - Network Selection Game. Fig. 3.2 reports the SW when the allocation is performed in the centralized setting, both optimally and using our proposed heuristic. Instead, in Fig. 3.6, we consider the *distributed* case and we show the SW when MCs reach the unique Nash Equilibrium of the network selection game we presented in Sec. 3.5. By comparing the value of the centralized optimal SW in Fig. 3.6a and 3.6b, we can state that in the distributed scenario we lose [1] up to 65% SW when $N = 0.0005$ (10^6 objects), and 60% when $N = 0.0512$ (10^4 objects), compared to the optimum (obtained with the ILP model (3.4)-(3.10)).

For the sake of brevity, we omit the results regarding the convergence speed of the best response dynamics for the network selection game. Indeed, in all the considered

1. Please note that this evaluation is done with respect to the social welfare SW, whereas in Sec. 3.5 we computed the Price of Anarchy with respect to the network congestion only.



(a) $N = 0.0005, |\mathcal{M}| = 60, P = 10$*USD Fun. of* $|\mathcal{A}|$ **(b)** $N = 0.0521, |\mathcal{M}| = 60, P = 10$*USD Fun. of* $|\mathcal{A}|$

Figure 3.6: Social Welfare - Network Selection Game*, The figures show a comparison of the Social Welfare (SW) obtained in the centralized (*Opt.*) and the distributed scenario (Nash Equilibrium,* NE*), in particular they show the performance loss due to the distributed allocation of the mobile clients. Fig. 3.6a represents the scenario with an object catalog of* 10^6 *($N = 0.0005$), while Fig. 3.6b refers to the case where the object catalog contains* 10^4 *objects ($N = 0.0521$), varying the number of APs.*

scenarios the game always reaches the equilibrium in less than 10 iterations.

3.7 Related Work

Incentive mechanisms and *auction theory* have been used to design efficient allocation mechanisms in several network contexts. In particular, as described in [45], they are extremely appealing to model, for example, the problem of leasing underutilized spectrum to secondary users, tackled in cognitive radio networks (e.g: [62, 107, 197]).

Vickrey auctions are used in [45] to make the cognitive radio users compete to acquire the radio resources; incentives are paid to the primary users for sharing their licensed spectrum. In [104] Jia et al. formulate a revenue maximization auction mechanism for a primary license holder, using the Vickrey-Clarke-Groves (VCG) mechanism. In [162] the authors propose to use a knapsack-based auction model to foster the dynamic spectrum allocation to secondary wireless service providers to maximize revenue and spectrum usage. Dynamic spectrum leasing is also the focus of [198], where secondary service providers lease spectrum from brokers to provide network connectivity to secondary users. Besides these classical scenarios, auctions are becoming an even more interesting tool to model bandwidth allocation. Dai et al. in [57] present a collaborative caching auction system for wireless video streaming based on the VCG mechanism. Their proposal fosters the cooperation between cache servers by ensuring that every bidder declares his real private valuation for the auctioned resource, thus ensuring the *truthfulness* property.

Meanwhile, the migration to the novel paradigm of NDN will greatly modify the nature of economic interactions between different parties at the network-wide scale. In particular, Rajahalme et al. have thoroughly described in [149] the changes that will likely incur at the inter-domain point of view, when updating the infrastructure from a

TCP/IP network to NDN. Pham et al. have modeled in [144] the economic interactions between the different actors of an Information Centric Network, using the framework of game theory. Game theory is also used in [96] to propose an incentive scheme to foster cache cooperation and users' collaboration for multimedia streaming purposes. In particular, the authors tackle the problem as a resource allocation game in both the cooperative as well as the non-cooperative settings.

Unlike the surveyed literature, our approach is to tackle the mechanism design problem, while jointly taking into account the available bandwidth as well as the storage space offered by every access point owner. Given the fact that we study the performance of this mechanism for NDN, the availability of caches not only reduces the backhaul capacity used at every access point, but it is also beneficial to the content provider himself, since it offloads its infrastructure.

Lastly, the problem of incentive mechanisms to remunerate AP owners is presented in [94, 95]. The focus of [95] is to consider the scenario in which mobile user equipments behave as wireless access point and provide network connectivity through 3G/4G connections. While in [94], the authors formulate a double auction for a fixed network.

3.8 Conclusion

This chapter proposed a novel mechanism for heterogeneous Information Centric Networks to stimulate wireless access point owners to jointly lease their unused bandwidth and storage space to a content provider, under the partial coverage constraints. By jointly leasing the spare bandwidth and caching storage offered by access points owners, the content provider can increase the number of mobile clients reached by his service, while offloading his distribution infrastructure by replicating the provided contents closer to the users' location, and therefore saving significant amount of OPEX costs and energy required by the distribution infrastructure.

We provided an algorithm to determine the optimal allocation of mobile clients to access points that ensures the *individual rationality* as well as the *truthfulness* properties by forcing the AP owners to bid the real valuation for the offered resources. We showed that the optimal allocation problem is NP-hard, and provided three efficient alternatives of a greedy algorithm that compute a sub-optimal solution of the problem in polynomial time, while still guaranteeing the individual rationality as well as the truthfulness properties.

We also accurately modeled a distributed, realistic setting, where the mobile clients autonomously select the access point that they should connect to, taking into account real channel properties and the MAC behavior of the 802.11 technology. Specifically, we modeled the interactions between different users as a non-cooperative network selection game, considering as utility function the achievable throughput in loaded conditions. Furthermore, we proved that the game exhibits desirable properties: 1. existence and convergence to the pure Nash equilibrium, since it is *weighted potential*; 2. uniqueness of one such Nash equilibrium and, 3. we also provide a lower bound on the price of anarchy.

Finally, numerical results demonstrated that the performance of the greedy algo-

rithms well approaches the results obtained by the optimal solution. In particular, in the worst case we considered, the heuristics computed a solution whose social welfare was only 16% lower than the optimal counterpart, while being several order of magnitudes faster that the optimal allocation algorithm.

Part III

Network Planning for Content Distribution

CHAPTER 4

Optimal Design of Information Centric Networks

The first prototypes for NDN-capable routers have already begun to appear, however, in order to migrate to this novel paradigm, network operators should make non-negligible investments to purchase the new NDN devices. On top of that, many research works have clearly shown that the adoption of NDN can foster a better content distribution: even by deploying relatively small caches, remarkable hit-rates can easily be experienced. On the other hand it is still unclear what is the real economic benefit that the adoption of one such technique can lead to network operators.

For these reasons, this chapter tackles the network planning problem for the migration to Named-Data Networking. In particular it provides clear quantitative insights of the economic benefits that operators can expect by switching to NDN. Our contribution is to shed lights on one such important question, by formulating a *content-aware network-planning* problem, as well as a novel optimization model to study the migration to NDN, in a budget-constrained scenario.

Our formulation takes into accurate account traffic routing and content caching. We prove that the optimization problem is NP-Hard, then we formulate heuristics to efficiently solve it. An extensive simulation campaign with real network topologies shows that our greedy heuristic cuts the computation time while finding close to optimal solutions, and therefore can effectively support network operators to evaluate the effects of a migration to NDN.

4.1 Introduction

The first working prototypes for NDN-enabled routers have already been realized by Alcatel [179], Cisco [169] and Parc [4]. Specific hardware and software components are required in order to support NDN packet forwarding at wire-speed, and thus operators will certainly have to make non-negligible investments to purchase the new NDN devices. As a result, they will be willing to transition their infrastructures to NDN only if clear economic benefits are envisioned: by upgrading a router to NDN and by installing a given amount of storage to memorize frequently requested contents, it will directly serve incoming requests for the cached objects. In this way, given the fact that the content popularity is very skewed (i.e: few objects generate most of the traffic [30, 142]), the operator can experience significant economic savings accountable to traffic offloading [194]. While CDNs may also be used to efficiently serve clients' requests, they are usually regarded as an expensive solution, since they demand to centrally orchestrate replica placement and request routing [81], while in NDN each network device will autonomously perform these choices, thus reducing the overall management costs.

To pave the way for a potential paradigm shift from a TCP/IP network to NDN, we specifically consider the *migration* step to the NDN architecture and we formulate a novel *content-aware network planning model* that we use to compute the optimal *migration strategy* for the operator. On top of that, by considering relevant economic parameters, our model can also be used to understand which economic benefits are expected as a result of the transition to NDN. To achieve all these objectives, we take into account three economic parameters: 1. a traffic-proportional link cost, 2. the router migration cost and 3. the storage cost, proportional to the amount of memory installed at a given NDN-migrated node.

To summarize, in this chapter we provide the following contributions:

1. We formulate a model to evaluate the optimal content-distribution performance of an IP network under unsplittable routing conditions.
2. We propose a novel *content-aware network-planning* Mixed Integer Linear Programming (MILP) model for the *migration* to NDN. Our formulation determines the optimal node migration and cache allocation in a budget-constrained scenario. Unsplittable routing conditions are still enforced by non-migrated routers.
3. We prove that the content-aware network-planning problem is NP-Hard, therefore we propose a novel and very efficient *greedy* heuristic, that outperforms the *randomized rounding* algorithm we designed in [126].
4. We compare the performance of the *randomized rounding* heuristic with the new *greedy* solver, showing that the latter dramatically improves the quality of the final solution while cutting the computation time of the heuristic of at least an order of magnitude.
5. We quantitatively evaluate the benefits of migrating to NDN, with different budgets as well as pricing configurations.

Our key findings suggest that 1. for a very large span of pricing policies, by migrating only few nodes to NDN remarkable traffic reductions will be experienced by the

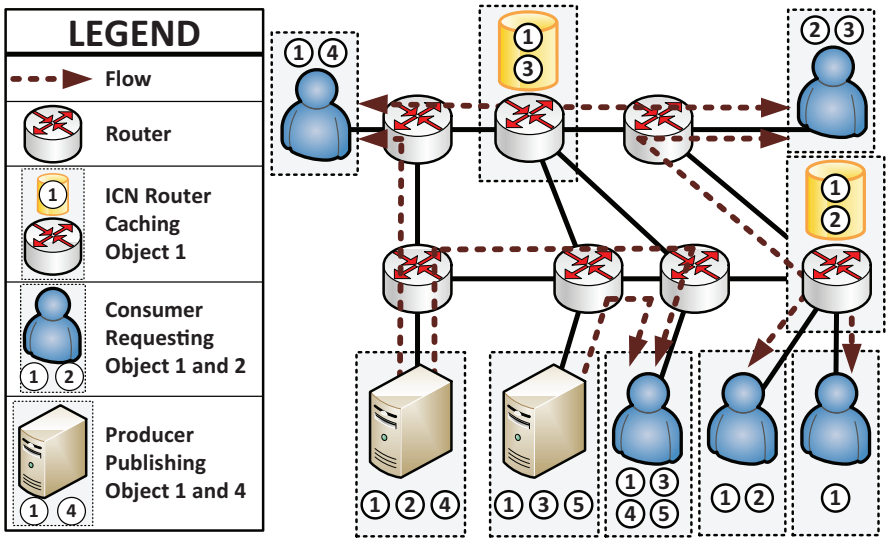


Figure 4.1: System model. *Given the network topology, consumers' requests, and objects served by content providers, our optimization model chooses which routers should be migrated to NDN, and which objects should they cache.*

operator; 2. NDN benefits also content providers since it significantly offloads their distribution infrastructures, and 3. when the content popularity distribution is very skewed (i.e: most of the traffic is generated by few popular objects), the storage space installed at the migrated nodes is an order of magnitude smaller than for less skewed distributions.

This chapter is structured as follows: in Sec. 4.2 we introduce the system model. In Sec. 4.3 we extensively describe the optimization models we use to compute the overall content delivery cost of an IP network and the content-aware planning model for the migration to an NDN. In Sec. 4.4 we illustrate our proposed *randomized rounding* and *greedy* heuristics for NDN, while numerical results are discussed in Sec. 4.5. Related works are presented in Sec. 4.6, and finally, concluding remarks are illustrated in Sec. 4.7.

4.2 System Model

In this section we describe the system model and discuss the rationale of our approach.

Fig. 4.1 represents an example to describe relevant features of our proposed system model. Three types of nodes are available in the topology: consumers, producers and routers. All the nodes operate on a finite set of contents, called the *"catalog"*. For the sake of simplicity, as depicted in Fig. 4.1, in this example we assume that the catalog is composed of 5 objects. Producers publish objects in the network, whereas consumers generate demands for them. It is possible that the same object is provided by different producers, as represented in the figure.

Each link in the network is characterized by having a traffic-proportional cost (OPEX)

and a bounded capacity. The network operator can significantly reduce the traffic costs by migrating routers to NDN and leveraging their in-network caching functionalities. However, to perform the migration, the operator must pay the corresponding costs (CAPEX) which are given by (1) the price to migrate a router to NDN, C^M and (2) the storage price to memorize one object at a migrated router, C^S. We bound the migration costs (i.e, those due to node migration and caching storage) to the value B, that is the total migration budget the operator is willing to spend. NDN routers issue upstream traffic requests as if they were the request origins; finally, on top of offering caching functionalities, they also support splittable request routing at the level of the single object requests. We assume that any router in the topology can be migrated to NDN, however it is very simple to extend our models and algorithms to restrict the set of routers eligible for the migration.

Intra-domain routing protocols such as OSPF provide equal-cost multipath (ECMP) functionalities, to support flow splitting over multiple paths having equal-cost weights [70]. However, in this work we formulate the optimization problem for an IP network by explicitly considering unsplittable routing conditions. There are many reasons behind this choice: first of all, to the best of our knowledge, network operators are often still reluctant to actually take advantage of multipath functionalities, since having to deal with single-path flows facilitates network management and troubleshooting [17, 22, 25]. Secondly, there are some scenarios in which avoiding packet reordering becomes a major requirement and therefore multipath flow splitting should be prevented [25]. On top of that, the focus of our contribution is to consider the overall content distribution costs expressed as a function of the link costs which are provided as input parameters and are fixed; on the other hand, in order to take advantage of ECMP functionalities, the link weights are dynamically changed by the operator as a function of the actual congestion. However, for the sake of completeness, in Sec. 4.5.6 we also take into account the case in which unsplittable traffic conditions are relaxed, showing that, on average, their impact on the overall cost is rather limited.

In this work we are interested in studying the performance bounds that can be achieved by migrating a subset of the available routers to NDN. In order to do that, we consider the *off-path* caching scenario, as shown in Fig. 4.2.

In particular, while in *on-path* caching flows are forwarded on the shortest path to the closest producer publishing the requested content, in *off-path* caching network nodes have a "full" visibility of the contents stored in each NDN router. Therefore, a node can eventually divert traffic requests on a path where a copy of the content can be retrieved, saving the cost to contact the original producer. In Fig. 4.2 a toy example showing this positive advantage is represented. In fact, while in on-path caching the overall cost [1] to serve traffic demands is 38 US$, in off-path caching only 23 US$ are required. On the other hand, the computational complexity of the object-placement problem is significantly increased due to the fact that object placement and routing must be jointly optimized network-wide.

In the rest of the chapter we describe our proposed optimization models and heuris-

1. As detailed in Sec. 4.3, the overall cost is given by the sum of the product of the forwarded traffic and the link cost.

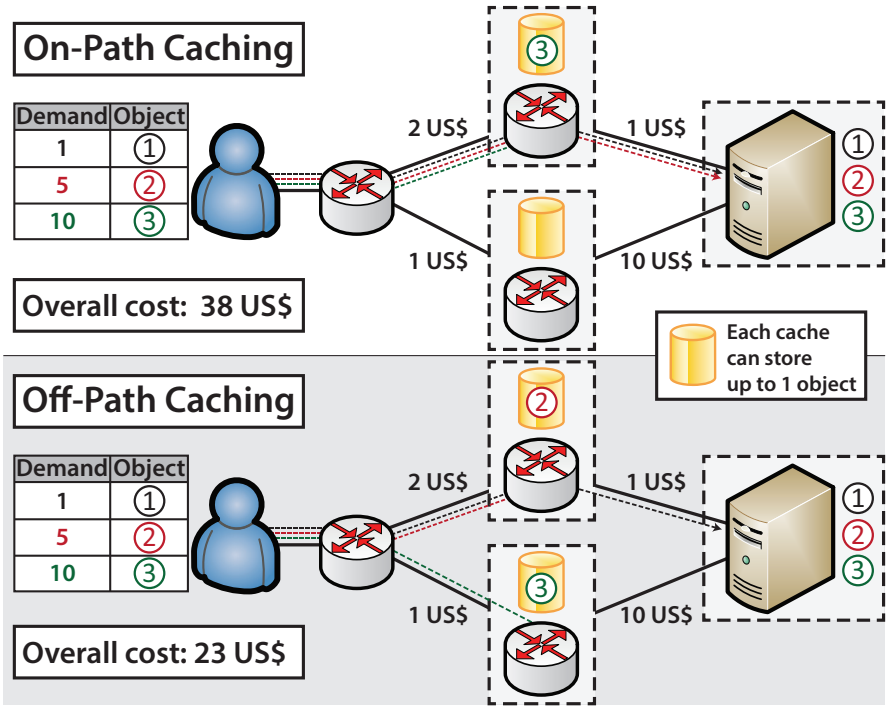


Figure 4.2: On-path vs. Off-path caching. *In the example, the upper path is the shortest to the producer (in terms of cost), moreover each router can store up to 1 object in its cache. The overall cost is the sum of the product of the traffic demand and the link cost on which flows are forwarded. In the* on-path approach *only the shortest path can be used to serve the demands and the minimum cost solution of 38 US$ is obtained by storing object* 3 *in the intermediate router. On the other hand, in the* off-path approach*, also the lower path can be used and the minimum cost solution improves to the lower value of 23 US$, that is achieved by placing object* 3 *and* 2 *in the caches of the lower and upper router, respectively.*

tics to help the operator determine the optimal NDN node migration strategy, object placement and request routing, considering off-path caching. We focus on this case since it is the one that leads to the best performance bounds, even though our models and algorithms can easily be extended to the on-path caching scenario.

4.3 Design Models

We now describe the optimization models we use to evaluate the migration to an NDN. Sec. 4.3.1 presents the IP network model, while Sec. 4.3.2 is devoted to the NDN network planning formulation, as well as the formal proof that the two problems are NP-Hard.

Let us introduce the notation used in describing the planning problems and in the optimization models. We represent the network as a directed graph $G = (N, A)$, where the set of nodes N is partitioned into consumers C, producers P, and routers R (i.e, $N = C \cup P \cup R$).

The set of forward and backward arcs of node $i \in N$ are denoted with $FS(i)$

Table 4.1: *Summary of the notation used in this chapter.*

Parameters of the Models	
A	Set of arcs
$N, C,$ P, R	N Set of nodes, $C \subset N$ Set of consumers, $P \subset N$ Set of producers, $R \subset N$ Set of routers
Q	Set of requesters. In IP $Q = C$, in NDN $Q = C \cup R$
O	Set of objects
$FS(i)$	Set of forward arcs $(i, j) \in A$ for node $i \in N$
$BS(i)$	Set of backward arcs $(j, i) \in A$ for node $i \in N$
$b_{i,j}$	Capacity of arc $(i, j) \in A$
$p_{i,j}$	Price per unit of traffic on arc $(i, j) \in A$
d_c^o	Demand of consumer $c \in C$ for object $o \in O$
r_p^o	0-1 Object reachability matrix. $r_p^o = 1$ if $p \in P$ can serve $o \in O$
C^M	Price to migrate one router to NDN
C^S	Price to install one unit of storage
B	Total migration budget

Decision Variables of the Models	
$y_{i,j}^{o,q}$	Flow on $(i, j) \in A$ for object $o \in O$, requested by $q \in Q$
$z_{i,j}^q$	0-1 Routing variable: $z_{i,j}^q = 1$ if $(i, j) \in A$ can route requests for $q \in Q$
m_r	0-1 Router migration variable: $m_r = 1$ if $r \in R$ migrates to NDN
k_r^o	0-1 Cache storage variable: $k_r^o = 1$ if $r \in R$ caches $o \in O$
w_l^o	Flow served by $l \in (P \cup R)$ for $o \in O$, when l stores a replica of o
$F_r^{o,q}$	Flow balance at router $r \in R$, for object $o \in O$, requested by $q \in Q$

and $BS(i)$, respectively. Network arcs $(i, j) \in A$ are characterized by a capacity, denoted with $b_{i,j}$, and a price per unit of traffic, $p_{i,j}$. This is representative of the prices charged by services like Amazon CloudFront [1], as we discuss in Sec. 4.5.1. We denote with O the set of objects, and we assume that all of them have the same size, as frequently done in the literature (e.g: [125, 185]). Let Q be the set of requesters; for both the IP and NDN network models, requesters are nodes from which traffic requests originate: in the IP network, only the consumers can behave as such, and thus $Q \equiv C$. Each consumer $c \in C$ expresses a traffic demand d_c^o for object $o \in O$. Producers can serve a subset of the entire object catalog, in particular we represent with the binary parameter r_p^o the object reachability matrix ($r_p^o = 1$ if producer $p \in P$ publishes object $o \in O$, otherwise $r_p^o = 0$). For the sake of clarity, in Table 4.1, we summarize the notation used throughout the chapter.

4.3.1 IP Network model

We start by describing the IP network model we use as a benchmark with respect to the solution we get when studying the planning of an NDN. In the IP routing problem, objects must be routed from producers where they are available to consumers, possi-

bly passing through routers, at the minimum overall cost. We assume that flows are unsplittable.

The problem can be naturally described as a multicommodity flow model, where a commodity is associated with every pair {object, requester}. Let variables $y_{i,j}^{o,q}$ denote the flow of object $o \in O$ on arc $(i,j) \in A$ for requester $q \in Q$. In addition, in order to account for the unsplittable flow requirement, we introduce binary variables $z_{i,j}^{q}$ whose value is 1 if arc $(i,j) \in A$ is used to route traffic for requester $q \in Q$. Note that in the presence of multiple copies of the same content, the selection of the producer to serve each request is accomplished with variables w_p^o denoting the flow of object o served by producer p.

The minimum cost request routing problem for an IP network can therefore be formulated as follows:

$$\min \sum_{(i,j)\in A} p_{i,j} \sum_{o\in O} \sum_{q\in Q} y_{i,j}^{o,q} \tag{4.1}$$

subject to:

$$\sum_{(j,r)\in BS(r)} y_{j,r}^{o,q} - \sum_{(r,j)\in FS(r)} y_{r,j}^{o,q} = 0 \qquad \forall o \in O, \forall q \in Q, \forall r \in R \tag{4.2}$$

$$\sum_{(j,i)\in BS(i)} y_{j,i}^{o,i} = d_i^o \qquad \forall o \in O, \forall i \in C \tag{4.3}$$

$$\sum_{q\in Q} \sum_{(p,j)\in FS(p)} y_{p,j}^{o,q} = w_p^o \qquad \forall o \in O, \forall p \in P \tag{4.4}$$

$$w_p^o \leq r_p^o \sum_{c\in C} d_c^o \qquad \forall p \in P, \forall o \in O \tag{4.5}$$

$$\sum_{c\in C} d_c^o = \sum_{p\in P} w_p^o \qquad \forall o \in O \tag{4.6}$$

$$\sum_{o\in O} \sum_{q\in Q} y_{i,j}^{o,q} \leq b_{i,j} \qquad \forall (i,j) \in A \tag{4.7}$$

$$\sum_{o\in O} y_{i,j}^{o,q} \leq b_{i,j} z_{i,j}^{q} \qquad \forall i \in N \setminus C, \forall (i,j) \in FS(i), \forall q \in Q \tag{4.8}$$

$$\sum_{(i,n)\in FS(i)} z_{i,n}^{q} \leq 1 \qquad \forall i \in N \setminus C, \forall q \in Q \tag{4.9}$$

$$z_{i,j}^{q} \in \{0,1\} \qquad \forall q \in Q, \forall (i,j) \in A \tag{4.10}$$

$$w_p^o \geq 0 \qquad \forall p \in P, \forall o \in O \tag{4.11}$$

$$y_{i,j}^{o,q} \geq 0 \qquad \forall o \in O, \forall q \in Q, \forall (i,j) \in A. \tag{4.12}$$

The objective function (4.1) minimizes the overall traffic costs incurred by the provider on all network arcs.

The flow balance at every router and consumer node is imposed by (4.2) and (4.3), respectively. The balance at producer nodes depends on the requested flow of each

object (4.4) which is regulated by (4.5) and (4.6). These constraints consider the fact that requests can be served only by those producers that are actually publishing a copy of the given object in the network, and that the overall traffic served by the producers equals the overall demand expressed by the consumers.

Capacity constraints are enforced in (4.7). Unsplittable routing conditions are imposed in (4.8) and (4.9). In particular, the set of constraints (4.8) makes sure that flows for requester $q \in Q$ are forwarded only on the arcs $(i,j) \in A$ where $z^q_{i,j} = 1$, whereas in (4.9) we make sure that routers and producers have at most only one egress arc for requester $q \in Q$.

Finally, non negativity on flow variables and binary condition on $z^q_{i,j}$ are imposed in (4.10)-(4.12). Notice that if 0-1 variables $z^q_{i,j}$ are fixed, the problem amounts to a continuous multicommodity flow that can be solved by standard linear programming solvers.

4.3.2 NDN Planning

We now extend the model presented in Sec. 4.3.1 to solve the *content-aware network planning* problem in NDN.

Let C^M denote the additional cost to migrate one IP router to NDN. Once that a router has been migrated to this paradigm, caching storage can be installed on it. C^S denotes the storage cost to add the caching space sufficient to memorize one object. The overall *migration cost* should not exceed the total available budget, which is denoted with B. Two sets of binary variables are used in the NDN planning model: m_r and k^o_p. They are such that $m_r = 1$ if router $r \in R$ migrates to NDN, otherwise $m_r = 0$; similarly $k^o_r = 1$ if router $r \in R$ caches object $o \in O$, while $k^o_r = 0$ if the object is not cached.

With $F^{o,q}_r$ we denote the flow balance at router $r \in R$ for object $o \in O$ requested by $q \in Q$. If $F^{o,q}_r > 0$ then $r \in R$ generates demands for object o and thus it is leveraging multipath routing; instead, if $F^{o,q}_r < 0$ the node behaves as a source node, serving traffic requests for the cached object o; finally if $F^{o,q}_r = 0$, r is only forwarding flows for o.

Given the above definitions we formulate the budget-constrained NDN planning problem (BC-NDN) as follows:

$$\min \sum_{(i,j)\in A} p_{i,j} \sum_{\substack{o\in O \\ q\in Q}} y^{o,q}_{i,j} + \left[C^M \sum_{r\in R} m_r + C^S \sum_{r\in R} \sum_{o\in O} k^o_r \right] \tag{4.13}$$

subject to (4.3)-(4.5), (4.7)-(4.8), (4.10), (4.12), and

$$\sum_{(j,r)\in BS(r)} y_{j,r}^{o,q} - \sum_{(r,j)\in FS(r)} y_{r,j}^{o,q} = F_r^{o,q} \qquad \forall o \in O, \forall q \in Q, \forall r \in R \quad (4.14)$$

$$-\sum_{q\in Q} F_r^{o,q} = w_r^o + \sum_{(i,r)\in BS(r)} y_{i,r}^{o,r} \qquad \forall o \in O, \forall r \in R \quad (4.15)$$

$$w_r^o \leq k_r^o \cdot \sum_{c\in C} d_c^o \qquad \forall o \in O, \forall r \in R \quad (4.16)$$

$$k_r^o \leq m_r \qquad \forall o \in O, \forall r \in R \quad (4.17)$$

$$y_{i,j}^{o,r} \leq m_r b_{i,j} \qquad \forall (i,j) \in A, \forall o \in O, \forall r \in R \quad (4.18)$$

$$\sum_{(i,j)\in FS(i)} z_{i,j}^q \leq 1 \qquad \forall i \in P, \forall q \in Q \quad (4.19)$$

$$\sum_{(r,j)\in FS(r)} z_{r,j}^q \leq 1 + m_r \cdot (|FS(r)| - 1) \qquad \forall r \in R, \forall q \in Q \quad (4.20)$$

$$\sum_{c\in C} d_c^o = \sum_{l\in P\cup R} w_l^o \qquad \forall o \in O \quad (4.21)$$

$$C^M \sum_{r\in R} m_r + C^S \sum_{r\in R} \sum_{o\in O} k_r^o \leq B \qquad (4.22)$$

$$y_{r,i}^{o,r} = 0 \qquad \forall r \in R, \forall (r,i) \in FS(r), \forall o \in O \quad (4.23)$$

$$w_l^o \geq 0 \qquad \forall l \in P \cup R, \forall o \in O. \quad (4.24)$$

The objective function (4.13) takes into account traffic and migration cost components. The former is given by $\sum_{(i,j)\in A} p_{i,j} \sum_{\substack{o\in O \\ q\in Q}} y_{i,j}^{o,q}$, the latter is instead the sum of node migration costs $C^M \sum_{r\in R} m_r$, and storage costs $C^S \sum_{r\in R} \sum_{o\in O} k_r^o$.

Flow balance constraints for routers are expressed in (4.14). In particular, if a router $r \in R$ migrates to NDN (i.e, $m_r = 1$), we let the flow balance be $F_r^{o,q} \leq 0$, meaning that r can directly serve incoming requests; otherwise, if $m_r = 0$ we set $F_r^{o,q} = 0$. The set of constraints (4.15) permits a router $r \in R$ to have caching functionalities (i.e, $w_r^o \geq 0$); furthermore it lets r behave as a requester (i.e, $y_{i,r}^{o,r} \geq 0$), a feature that facilitates traffic splitting in the network. The joint presence of constraints (4.16)-(4.18) makes sure that only NDN-migrated routers can provide caching functionalities and behave as requesters. In-network caching features are modeled in (4.16) and (4.17). In particular, if a router r migrates to NDN and stores in its local cache a copy of object o, it is then capable of directly serving upstream requests for that particular object. Instead, in (4.18) we prevent non-migrated routers to behave as requesters.

Unsplittable request routing is enforced in the set of constraints (4.19) (*for producers only*) and (4.20) (*for routers only*). In particular, this latter set of constraints lets a

migrated NDN router $r \in R$ support splittable routing: if the router does not migrate to NDN (i.e, $m_r = 0$), then at most one egress link is used to route requests for $q \in Q$, otherwise if $m_r = 1$, then all the egress links can be used, making the NDN-migrated router capable to perform multipath routing. In (4.21), we impose the condition that the overall demand generated for object $o \in O$ is satisfied by producers and caching routers. The budget allocated for the migration is limited by (4.22). The set of constraints (4.23) avoids loops, preventing requests expressed by a router to be fulfilled by the router itself, while in (4.24) non-negativity on flow variables is enforced.

We now study the computational complexity of our proposed BC-NDN problem and we formally prove that it is NP-Hard.

Theorem 5. *The budget-constrained NDN planning problem (BC-NDN) is NP-Hard.*

Proof. In the *single-source unsplittable flow problem* (UFP) we want to find a feasible unsplittable routing for all the commodities of a network $G = (V, E, u)$, given a source vertex $s \in V$, a set of k commodities with sinks $t_1, \ldots, t_k$ and a corresponding real-valued demand $\rho_1, \ldots, \rho_k$. The unsplittable routing condition is enforced by routing the ρ_i demand on a single $s - t_i$ flow path. The feasibility question for UFP is strongly NP-complete [110].

Consider any instance of UFP. We polynomial-time reduce it to BC-NDN as follows: first of all we set unitary link-prices, as well as $C^M = 1, C^S = 1, B = 0$. We then create a new network with one producer, corresponding to the single source vertex $s \in V$ for UFP, and k consumers, one for each of the commodities available. The object catalog is composed of the k commodities (i.e, $|O| = k$). Consumers' demands are set as follows:

$$\begin{cases} d_c^o = \rho_c & \forall c \in C, o \in O \;\; o = c \\ d_c^o = 0 & \forall c \in C, o \in O \;\; o \neq c \end{cases} \tag{4.25}$$

Since we reduced UFP to BC-NDN in polynomial-time, BC-NDN is NP-Hard. □

Furthermore, it is easy to show that UFP can be reduced in polynomial-time to the optimal planning problem of the IP network we presented in Sec. 4.3.1, therefore we conclude that both optimization problems are NP-Hard.

4.4 Heuristics

The network planning problems we introduced in the previous section are NP-Hard and, as we will show in Sec. 4.5, even by using best of breed ILP solvers available today, it is often very challenging, in terms of computation time, to find the optimal NDN planning solution (as we discuss in Sec. 4.5.3). Therefore, there is the clear need to formulate heuristics that can efficiently find a close to optimal solution for the network planning problem. While the computation time is not a concern per-se (since we are going to run the algorithms off-line), our main goal while formulating the heuristics is to be able to solve very large network instances. To this aim, in the following Sec. 4.4.1 we briefly describe the randomized rounding heuristic we originally formulated in our

Algorithm 5: Randomized Rounding

Input : $mdl \Leftarrow \langle b_{i,j}, p_{i,j}, d_c^o, r_p^o, C^M, C^S, B \rangle$
Output: obj_fun
1 $\widehat{k}_r^o \Leftarrow$ SolveRelaxedNDNModel(mdl);
2 $\widehat{k}_r^{max} \Leftarrow \max_{\forall o \in O} \widehat{k}_r^o$;
3 $RL \Leftarrow$ SortRoutersByCumulativeProbabilityPerObject$(\widehat{k}_r^o)$;
$C \Leftarrow 0$;
foreach $r \in RL$ **do**
  $\bar{m}_r \Leftarrow false$;
  foreach $o \in O$ **do**
4 $w \Leftarrow \left\{ \text{UniformRndValue}(0,1) \leq (\widehat{k}_r^o / \widehat{k}_r^{max}) \right\}$;
5 **if** $w \wedge (C < B)$ **then**
6 **if** $\neg\, \bar{m}_r$ **then** $C \Leftarrow C + C^M$;
7 $C \Leftarrow C + C^S$; $\bar{k}_r^o \Leftarrow true$; $\bar{m}_r \Leftarrow true$;
  end
end
8 $obj\_fun \Leftarrow$ SolveNDNModel$(mdl, \bar{k}_r^o, \bar{m}_r)$;

previous work [126]; then in Sec. 4.4.2 we illustrate our novel greedy heuristic that outperforms the randomized rounding algorithm, as we extensively show in the numerical comparison in Sec. 4.5.

4.4.1 Randomized Rounding Heuristic

Algorithm 5 illustrates the randomized rounding heuristic in pseudo-code. The rationale behind it is to solve the continuous relaxation of the NDN model described in Sec. 4.3.2, computing the optimal fractional values of $\widehat{k}_r^o$. We then interpret $\widehat{k}_r^o$ as the probability that object $o \in O$ is placed in the cache of the migrated router $r \in R$. As frequently done in the randomized rounding literature [109], we scale the relaxed variables for object caching $\widehat{k}_r^o$ dividing them by $\widehat{k}_r^{max}$, in order to increase the object caching probability. Then, we assign a value to the suboptimal binary variables $\bar{k}_r^o$, setting them to one with a probability equal to $\widehat{k}_r^o$. As a result, the algorithm chooses the node migration $\bar{m}_r$ and object caching variables $\bar{k}_r^o$.

More specifically, if we refer to Alg. 5, the solution of the continuous relaxation of the NDN model is computed in Step 1. In Step 2, the algorithm extracts $\widehat{k}_r^{max}$, that is the largest $\widehat{k}_r^o$ value for each router. Instead, the cumulative caching probability for all the objects (i.e, the value $\sum_{\forall o \in O} \widehat{k}_r^o$) is computed in Step 3, where we also sort the routers in non-increasing order according to such metric. The overall migration costs are denoted by C. In Steps 4 and 5, the randomized caching choice is performed: the algorithm caches object $o \in O$ at router $r \in R$ with probability $\widehat{k}_r^o / \widehat{k}_r^{max}$ if and only if there exists sufficient spare budget. In Step 6 the node migration costs are added to the

Algorithm 6: Greedy Node Migration Algorithm

Input : $mdl \Leftarrow \langle b_{i,j}, p_{i,j}, d_c^o, r_p^o, C^M, C^S, B \rangle$
Output: obj_fun

```
1 spareBudget ⇐ B;
2 while ∃r ∈ R|RouterMigrationSavings(r, mdl) > 0∧ spareBudget > C^M do
3 |   arg max_{r∈R} RouterMigrationSavings(r, mdl) ;
4 |   r.migrate();
5 |   cachedObjects ⇐ r.numOfCachedObjects();
6 |   spareBudget ⇐ spareBudget −C^M − C^S· cachedObjects;
7 |   OffPathCachingMulticommodityFlowAlgorithm(mdl);
  end
```

value of C, while in Step 7, the storage costs are included and the caching variables are set. Finally, in Step 8, we solve the NDN model by fixing the caching and migration variables.

4.4.2 Greedy Heuristic

In this section we extensively describe the novel greedy heuristic we propose for the planning problem.

The rationale behind the greedy heuristic is to iteratively migrate, the "*most promising*" router, considering the benefits that can be achieved if (1) traffic requests are sent on the shortest path towards the closest *publisher* (i.e, a producer or an NDN-router storing a copy of the requested content) and (2) the router caches content objects and directly serves incoming traffic requests, thus offloading the networking infrastructure. As we will extensively discuss in the numerical results (Sec. 4.5), the greedy heuristic outperforms the randomized rounding with respect to both the computation time, as well as the quality of the final solution computed. Our heuristic is composed of three functions: the *"Greedy Node Migration"* (illustrated in Alg. 6) is the main function and it invokes the two sub-functions *"Router Migration Savings"* (illustrated in Alg. 7), and *"Off-Path Caching Multicommodity Flow"* (illustrated in Alg. 8). In the rest of this section we extensively describe these three algorithms.

Greedy Node Migration. Alg. 6 shows the steps of this procedure. In Step 1, the spare budget variable is initialized to the B value. Then, in Step 2, whenever the spare budget is larger than the router migration cost C^M, and there exist a router $r \in R$ whose migration leads to positive savings (computed with Alg. 7), we migrate one router to NDN. In particular, in Step 3 we perform the greedy choice to select the router that maximizes the savings, while in Steps 4-6 we update the spare budget according to the node migration and cache storage costs. Network flows are routed in Step 7 using the off-path caching multicommodity flow algorithm, as in Alg. 8.

Considering off-path caching has a remarkable effect on both the way we choose the "most promising" router, as well as the way we actually route the flows. In particular, in Alg. 7 we compute the cost savings that the operator can experience by migrating

Algorithm 7: Router Migration Savings Algorithm

Input : $r, mdl \Leftarrow \langle b_{i,j}, p_{i,j}, d_c^o, r_p^o, C^M, C^S, B \rangle$
Output: savings

```
1  costs ⇐ C^M;
2  savings ⇐ 0;
   foreach o ∈ O do
3      cacheSavings ⇐ 0;
       foreach c ∈ C do
4          sp ⇐ GetShortestPathToClosestPublisher(c, o, p_{i,j});
5          ct ⇐ GetPathCost(sp);
6          spr ⇐ GetShortestPath(c, r, p_{i,j});
7          cr ⇐ GetPathCost(spr);
           if cr < ct then
8              cacheSavings ⇐ cacheSavings + (ct − cr) · d_c^o;
           end
       end
9      if cacheSavings > C^S ∧ costs + C^S < B then
10         savings ⇐ savings + cacheSavings;
11         costs ⇐ costs + C^S;
       end
   end
12 savings = savings-costs;
```

a given router to NDN, whereas flows are routed according to Alg. 8, as we describe below.

Router Migration Savings. The algorithm that computes the router migration savings is shown in Alg. 7. Among the input parameters we provide r, which is the router for which we want to compute the migration savings. The first steps of the algorithm are to initialize the costs and savings variables (Steps 1-2). Then, we compute the overall savings we can achieve by migrating router r to NDN. In particular, we want to store in the cache of router $r \in R$, all the objects that reduce the retrieval cost for the network operator. In order to do that, we compute in Step 4 the shortest path from consumer c to the closest publisher (either a producer, or a migrated NDN router caching object o), and in Step 5 we compute its cost. Similarly, in Steps 6-7 we compute the path and cost from c to router r, and in Step 8 we increment the cache savings if consumer c is closer to r than the original publisher it was using. After summing the benefits for all the consumers, in Steps 9-11 we choose whether it is worth to cache object o at router r, and eventually we update the value for the savings, as well as the storage costs. Finally, in Step 12 we update the actual savings, by jointly taking into account the storage as well as the nodes migration costs.

Off-Path Caching Multicommodity Flow. In Alg. 8 we route network flows considering off-path caching. For each object o requested by a consumer c, the algorithm computes the overall traffic costs by finding in Step 2 the closest publisher (either a pro-

Algorithm 8: Off-Path Caching Multicommodity Flow Algorithm

Input : $mdl \Leftarrow \langle b_{i,j}, p_{i,j}, d_c^o, r_p^o, C^M, C^S, B \rangle$
Output: trafficCost
1 trafficCost $\Leftarrow 0$;
foreach $c \in C$ **do**
  foreach $o \in O$ **do**
2 $sp \Leftarrow$ GetShortestPathToClosestPublisher($c, o, p_{i,j}$);
3 ForwardFlowOnShortestPath(c, o, d_c^o, sp);
4 trafficCost $\Leftarrow$ trafficCost + $d_c^o \cdot$ GetPathCost(sp);
  end
end

ducer, or an NDN router caching o) and forwarding the traffic demand on the shortest path towards this destination (Step 3). Link capacity conditions are enforced in Step 3: the function computes the shortest path on the residual capacity graph and allocates flows according to the spare link resources.

For the sake of completeness, we want to remark the fact that the procedure *GetPathCost* returns the cost of the path that it receives as input parameter, by summing all the $p_{i,j}$ values on the given path. Furthermore, the procedure *GetShortestPathToClosestPublisher*, receives as input parameters the consumer c, the object it is requesting o, as well as the set of link prices $p_{i,j}$; it then returns as output parameter the shortest path from c to the closest "*publisher*" that can be either a producer publishing o, or an NDN router caching a copy of the requested object.

4.5 Numerical Results

In this section we present the numerical results obtained by performing extensive analysis using our *content-aware network planning models* and the corresponding *randomized rounding* and *greedy* heuristics. In particular, in Sec. 4.5.1 we describe the parameters we used for the numerical analysis, while Sec. 4.5.2 shows the results obtained in an example scenario. In Sec. 4.5.3 we discuss the computational performance of the proposed algorithms. Sec. 4.5.4 shows the effect of the budget parameter, while Sec. 4.5.5 presents the sensitivity analysis to different pricing policies. Finally in Sec. 4.5.6 we discuss the effect of unsplittable routing conditions.

4.5.1 Parameters and Assumptions

Five real topologies have been considered: Netrail (7 nodes), Abilene (11 nodes), Claranet (15 nodes), Airtel (16 nodes) and Géant (27 nodes) [10]. We uniformly distribute 5 producers and 10 consumers in the network, connecting them at most to one router. All network links have a capacity of 10 Gbit/s, and each consumer generates an aggregate demand of 1 Gbit/s randomly distributed on the object catalog according to the Zipf popularity distribution [76]. Two Zipf alpha exponents have been considered

(as in [76]): $\alpha = 1.2$ is used to model a very skewed popularity distribution where few objects are frequently requested, whereas $\alpha = 0.8$ better represents less skewed demands. The object catalog is composed of 10^8 different packet *chunks* of 4kB each, as in [39, 77]. For scalability reasons, and as frequently done in the NDN literature [39], we aggregate the traffic demands on 100 popularity classes; in other words, we solve the planning problem setting $|O| = 100$. We further assume that traffic demands are expressed by the users for a mid-term timespan of one year, and thus 37 Pbytes will be transferred by the network to the consumers.

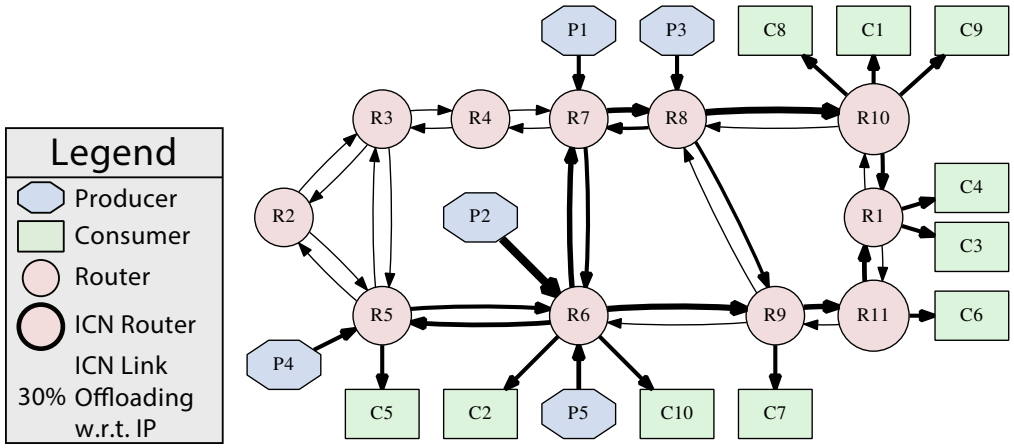
To transfer 1 Gbyte of data, Amazon nowadays charges a variable price in the range $[0.02; 0.085]$ USD [1]. Given such pricing, if 1 Gbit/s is constantly transferred on a link, its yearly cost will be in the range $[79k; 197k]$ USD; therefore, we uniformly generate the link price values ($p_{i,j}$) accordingly. Let $\max_p = 197k$ USD be the maximum yearly cost that the operator has to pay to satisfy the consumer's demand. We assume the cost to install one unit of storage is equivalent to $1/100$ of the yearly traffic cost, (i.e, $C^S = 0.01 \max_p$), and similarly we set $C^M = \max_p$ for the router migration. Finally, we assume that the total migration budget B is in the range $B \in [1; 7] \max_p$, and therefore we let at most 7 nodes migrate [2]. For each analysis, we performed 50 different runs and we computed the 95% confidence intervals depicted in the figures.

4.5.2 Example Scenario

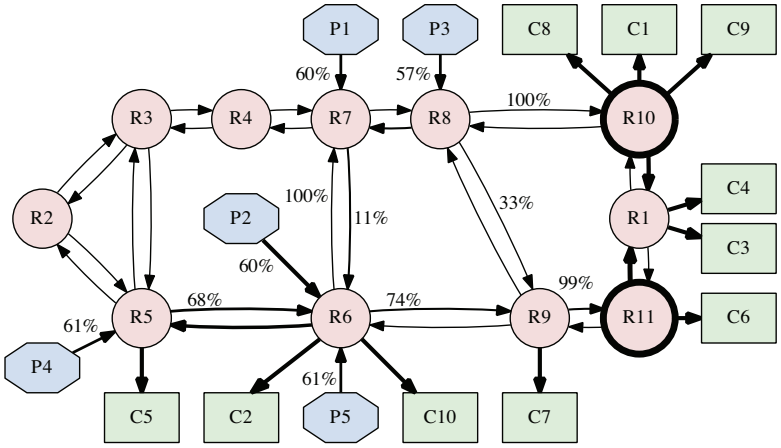
Fig. 4.3b represents the solution obtained considering the Abilene network topology for the IP network model. Despite the fact that this result refers to a Zipf with $\alpha = 0.8$, producers have a remarkably different load; in particular the first and the second most popular objects are published by producer P2 and P5, respectively, thus their links are the most congested. Fig. 4.3c, instead, represents the solution for the corresponding example migrated to NDN.

By comparing the solution depicted in Fig. 4.3b and Fig. 4.3c, we observe that migrating to NDN leads to an overall cost of $7.6 \cdot 10^6$ USD, compared to $14.3 \cdot 10^6$ USD for the IP network. Furthermore, such migration also reduces the link and producer congestion: in fact, on average, producers in NDN are providing 40% less traffic than in the IP solution. In general, network links are much less congested thanks to the presence of the two caches installed at routers R10 and R11. In addition, there exist some arcs, such as the one between R6 and R7, that are not carrying traffic anymore. Another interesting observation is that while router R10 is the one that is serving the highest number of consumers, router R11 is preferred by the model over R1. Therefore, to fully take advantage of the benefits introduced by NDN, we need to have an adequate network planning model to find the best network planning strategy. In fact, to find out the optimal migration strategy we cannot only take into account the number of consumers a router is serving, but we must have an adequate planning model such as the one proposed in this chapter.

2. This is done to make sure that at most all the nodes in the Netrail topology can migrate.



(a) *Legend* **(b)** *IP Network*

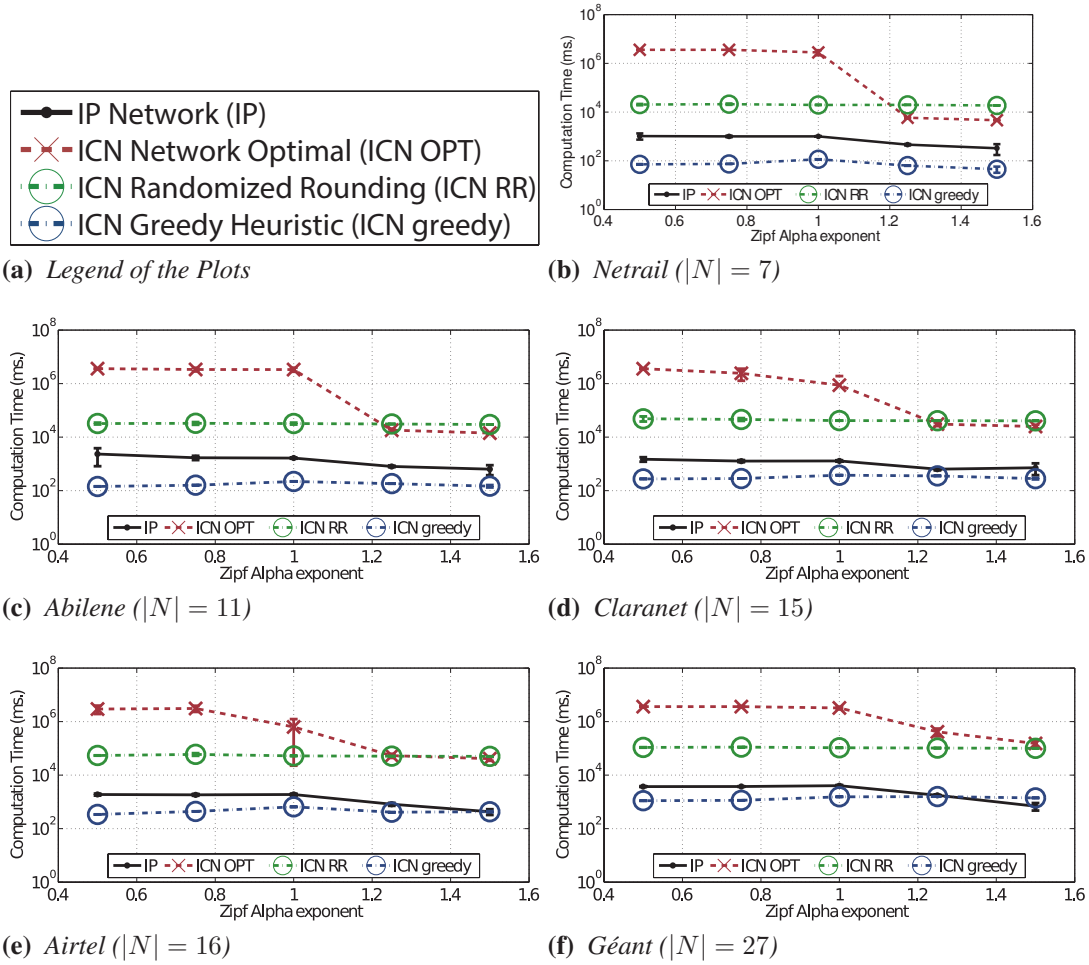


(c) *Optimal NDN Network*

Figure 4.3: Example Scenario. *We consider the Abilene topology with* $\alpha = 0.8$ *and budget of* $B = 3.5 \max_p$. *Fig. 4.3b represents the IP network, while Fig. 4.3c is the solution of the NDN planning model. The cost for the IP network is* $14.3{\cdot}10^6$ *USD, while in the optimal NDN network it decreases to* $7.6{\cdot}10^6$ *USD. Line size is proportional to the traffic the link is transferring, while the offloading percentage reported on each link in Fig. 4.3c compares the traffic in NDN w.r.t. the one of the IP network.*

4.5.3 Computing Time

Figures 4.4 report the average computing time necessary to solve different instances of our planning problem using the CPLEX 12.5 solver on a Dual Intel Xeon E5-2630 v2 @ 2.60GHz machine with 64 GByte of RAM. The figures refer to the scenario with budget $B = 7 \max_p$, and the scale is logarithmic. The common legend used in all the plots we are going to discuss hereafter is reported in Fig. 4.4a. For the optimal NDN model (NDN OPT), we set a maximum computation time of 1 hour and a relative MIP gap tolerance between the best integer objective and the objective of the best LP relaxation to 1%. As shown in Fig. 4.4, the solution of NDN OPT is strongly dependent on



(a) *Legend of the Plots*

(b) *Netrail* ($|N| = 7$)

(c) *Abilene* ($|N| = 11$)

(d) *Claranet* ($|N| = 15$)

(e) *Airtel* ($|N| = 16$)

(f) *Géant* ($|N| = 27$)

Figure 4.4: Computation Time Plots for Different Topologies. *Fig. 4.4a is the common legend we use for all the plots in Fig. 4.4-4.8. Fig. 4.4b-4.4f show the computation time for different topologies, using different algorithms, as a function of the Zipf α popularity exponent. The proposed greedy heuristic (NDN greedy) outperforms both the optimal MILP solver (NDN OPT), as well as the randomized rounding heuristic (NDN RR) cutting the execution time from hours to just few milliseconds, as shown in logarithmic scale in Fig. 4.4b-4.4f.*

the value of the Zipf popularity exponent α, as well as the size of the topology $|N|$. In particular, we observe that for large topologies, as well as small α values, more computation time is necessary to find the optimal solution of the planning problem. This behavior is caused by the fact that for small Zipf α exponents, the content popularity is less skewed, and therefore the object placement procedure needs to take into account also the possibility to cache objects that are not very popular. On the other hand, for large α values, the optimal solution can easily be found by the solver: the cached objects are only the very popular ones. The size of the topology $|N|$ has a similar impact on the computing time metric: the higher the number of nodes in the topology, the higher the number of combinatorial choices that the solver must take into account.

In all Figures 4.4b-4.4f there is a clear difference in computation time between the optimal solution for the NDN network obtained with the MILP solver (NDN OPT) and those obtained using the heuristics (NDN RR and NDN greedy): while the MILP solver for the optimal NDN network is very sensitive to the α parameters, the other trends are almost constant and independent from it.

We also observe that the greedy solver always outperforms of at least an order of magnitude the randomized rounding heuristic, often finding a solution in just few milliseconds rather than hours, as required by the optimal MILP solver. The greedy heuristic can scale to very large network instances, as a matter of fact, it can solve the Cogent topology (which is composed of 197 nodes), in less than 50 seconds, while instead we could not find the optimal solution using the MILP solver, in the same scenario.

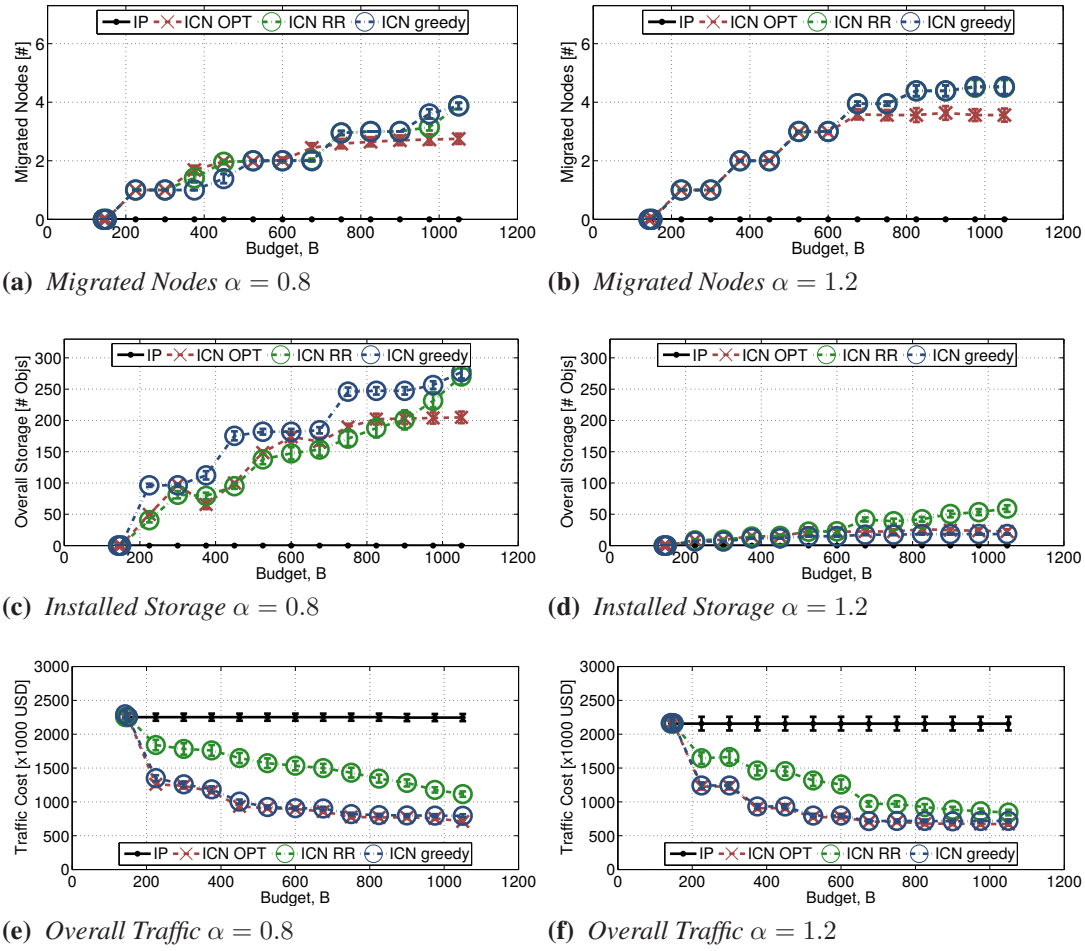
Despite the fact that the computation time is a very important parameter to evaluate the performance of a heuristic algorithm, we must also consider the quality of the final solution obtained. In the following, we show that the greedy heuristic outperforms the randomized rounding even with respect to the quality of the solutions, finding a very close to optimum value, when compared to the MILP solver.

4.5.4 Effect of the Budget

Fig. 4.5-4.6 show the effect of the available budget for the Abilene and the Géant topology, respectively. The horizontal line represents the reference value of the IP model, where no router can migrate to NDN. The number of migrated nodes in the Abilene topology with $\alpha = 0.8$ or 1.2 is shown in Fig. 4.5a and 4.5b. For both scenarios, the randomized rounding and greedy heuristics deploy more NDN routers, especially when the available budget is large; in particular, on average, they migrate 14% more routers than the optimal solution, for $\alpha = 1.2$. In the Abilene network, at most 5 nodes are migrated to NDN, and the larger the α, the higher the number of migrated nodes. However, as shown in Fig. 4.5c and 4.5d, the amount of storage deployed is strongly dependent on the α value. In particular, on average, when $\alpha = 1.2$ the optimal solution of the NDN planning problem installs 85% less storage than for $\alpha = 0.8$. In other words, for higher α values, it is better to deploy more nodes in the network rather than increasing their storage, while the opposite holds for smaller α. Observing Fig. 4.5c, we notice that the greedy heuristic slightly overprovisions the caching storage installed at the nodes.

Figures 4.5e and 4.5f show the traffic cost component for the Abilene topology. In both of them there is a steep decrease in cost when the budget goes from 1 to $1.5 \max_p$. On the other hand, for a larger budget, limited improvements are observed. In terms of the quality of the computed solution, the greedy heuristic is practically overlapping the optimal solution for both Zipf α values, whereas the randomized rounding approach introduces a small approximation, and it is slightly closer to the optimum when $\alpha = 1.2$, where it finds a solution that is on average only 16% more expensive than the optimal counterpart.

Combining the key findings we observed for the computation time, as well as those reported here concerning the quality of the computed solution, we conclude that our



(a) *Migrated Nodes* $\alpha = 0.8$ **(b)** *Migrated Nodes* $\alpha = 1.2$

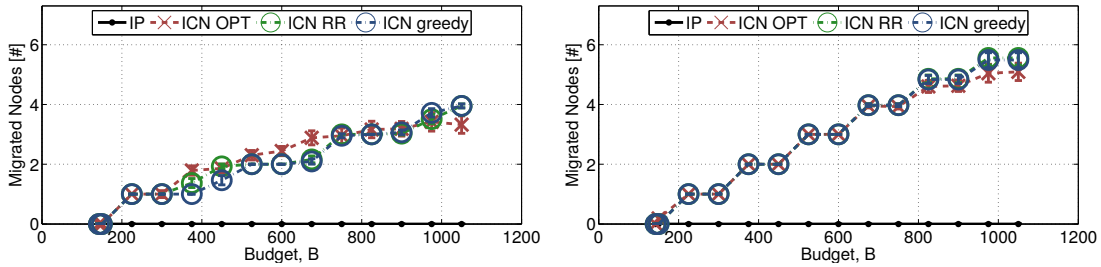
(c) *Installed Storage* $\alpha = 0.8$ **(d)** *Installed Storage* $\alpha = 1.2$

(e) *Overall Traffic* $\alpha = 0.8$ **(f)** *Overall Traffic* $\alpha = 1.2$

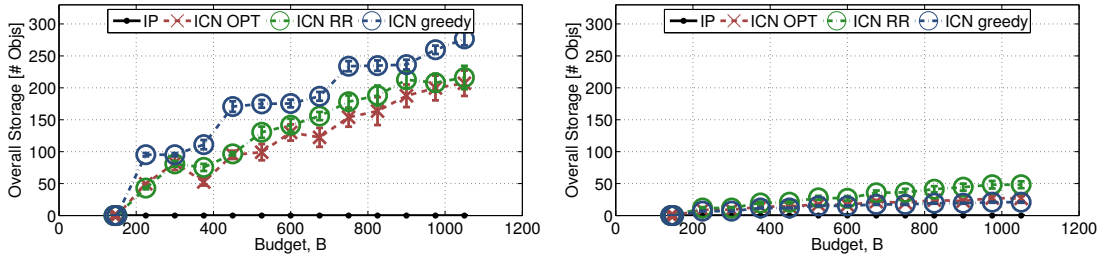
Figure 4.5: Budget Plots, Abilene Topology. *Fig. 4.5a-4.5b show the number of migrated nodes in the Abilene topology, as a function of the migration budget,* B*. In Fig. 4.5c-4.5d, we represent the amount of caching storage installed, while in Fig. 4.5e-4.5f the overall traffic cost is shown.*

novel greedy heuristic is to be preferred to the randomized rounding with respect to both the quality of the solution as well as the computation time.

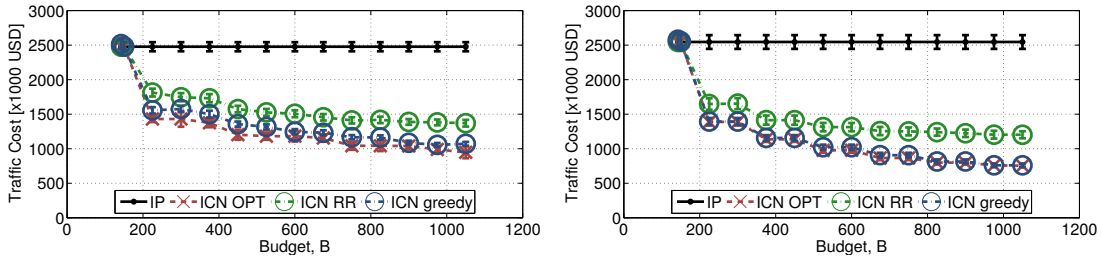
The Zipf popularity exponent has a negligible impact on the cost of the IP network, since it only affects traffic demands for single objects, but not their aggregate value. On the other hand, as expected, the network topology (in particular its diameter) has an remarkable effect on the overall costs, in fact, by comparing the overall cost of the IP network in the two topologies, we can conclude that, on average, Géant leads to a solution 10% more expensive than Abilene. This difference is even more remarkable, especially when considering smaller topologies; for instance, Géant is 48% more expensive than Netrail. In Géant, when $\alpha = 0.8$, the randomized rounding heuristic leads to solutions which are, on average, only 16% more expensive than their optimal counterparts. However, the greedy heuristic can do even better, lowering this gap to less than 5%. In line with previous literature [125, 185], NDN allows the operator to reduce



(a) *Migrated Nodes* $\alpha = 0.8$ **(b)** *Migrated Nodes* $\alpha = 1.2$



(c) *Installed Storage* $\alpha = 0.8$ **(d)** *Installed Storage* $\alpha = 1.2$



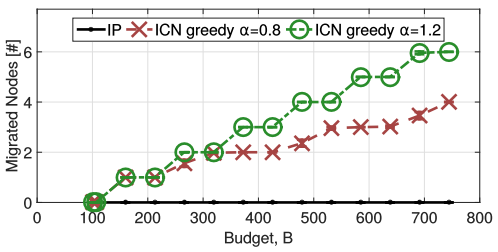
(e) *Overall Traffic* $\alpha = 0.8$ **(f)** *Overall Traffic* $\alpha = 1.2$

Figure 4.6: Budget Plots, Géant Topology. *Fig. 4.6a-4.6b show the number of migrated nodes in the Géant topology, as a function of the migration budget. In Fig. 4.6c-4.6d, we represent the amount of caching storage installed, while in Fig. 4.6e-4.6f the overall traffic cost is shown.*

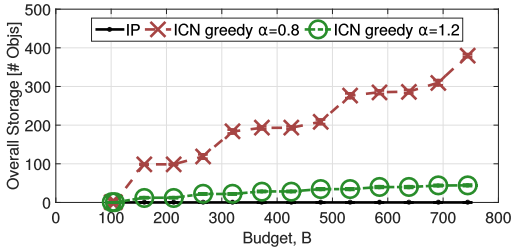
his traffic costs remarkably, even when the migration budget is very constrained, saving up to 68% of the overall traffic costs, as we observed in Géant, with $\alpha = 1.2$.

For the sake of completeness, Figures 4.7 show the effect of the budget in the Cogent topology. Since this topology is composed of 197 nodes, we could only run the model for the IP network and our proposed heuristic for NDN. Consistently with the trends obtained in the other topologies, in Fig. 4.7a we observe that when $\alpha = 1.2$ a higher number of nodes is migrated to NDN, although the amount of storage deployed is an order of magnitude smaller than the one used when $\alpha = 0.8$, as shown in Fig. 4.7b.

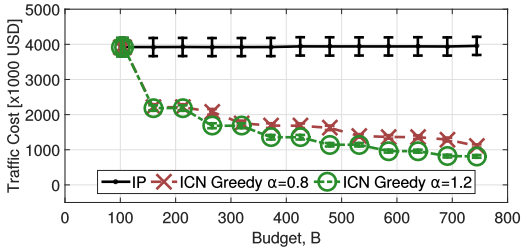
In terms of the overall traffic generated, as shown in Fig. 4.7c, in the Cogent topology we observe a similar trend to the one of Abilene and Géant. As expected, since the number of network nodes in Cogent is higher than that of Géant and Abilene, the overall traffic cost for the IP network is 38% higher than Géant, and 45% higher than Abilene. Furthermore, when considering the largest budget possible, the NDN solution



(a) *Migrated Nodes*



(b) *Installed Storage*



(c) *Overall Traffic*

Figure 4.7: Budget Plots, Cogent Topology. *Fig. 4.7a shows the number of migrated nodes in the Cogent topology, as a function of the migration budget. In Fig. 4.7b, we represent the amount of caching storage installed, while in Fig. 4.7c the overall traffic cost is shown.*

in Cogent is 23% and 33% more expensive (in terms of traffic costs) than Géant and Abilene, respectively.

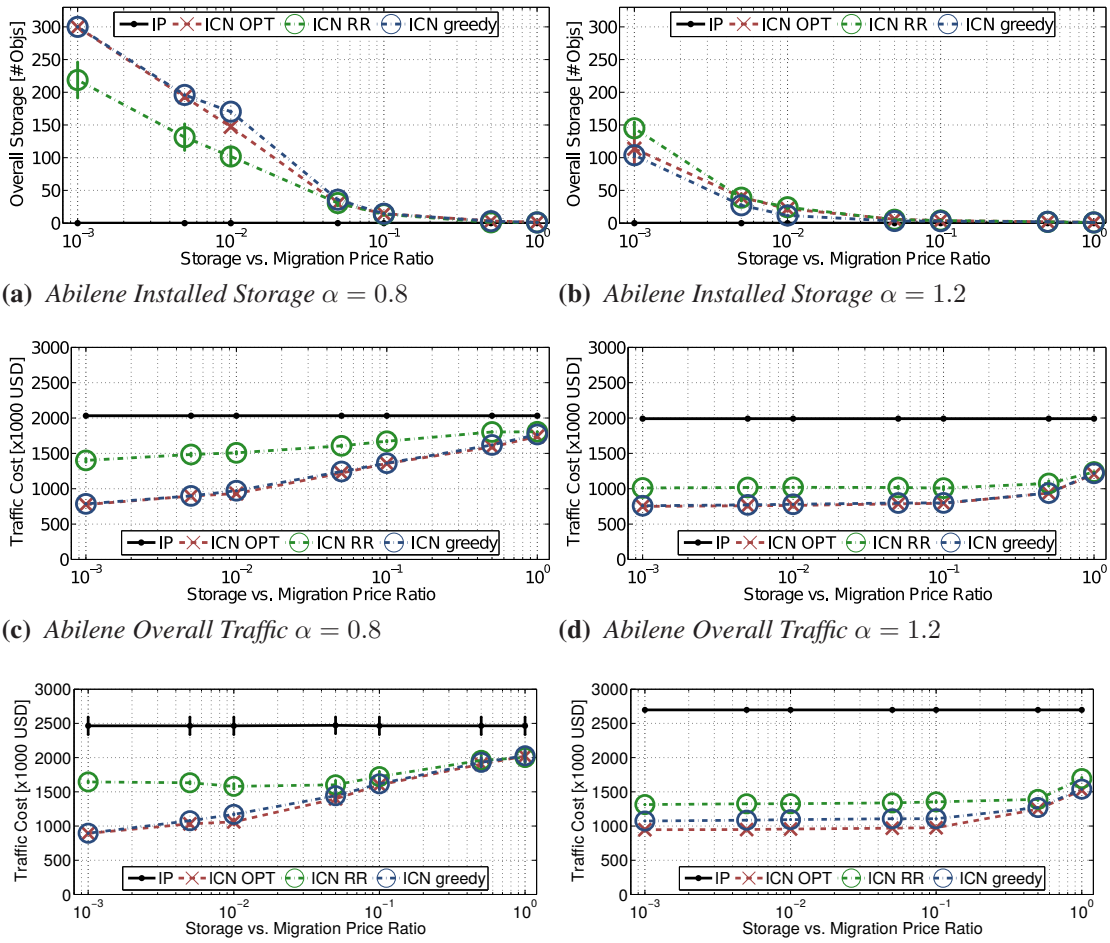
4.5.5 Effect of the Price

Figures 4.8 show the effect of different pricing policies for different topologies and Zipf α values. On the x-axis in Fig. 4.8 we report the *storage vs. migration price ratio* which is defined as $\frac{C^S}{C^M}$, where we assumed that $C^M = \max_p$ is fixed.

Fig. 4.8a-4.8b report the storage trend as a function of the price ratio with respect to $\alpha = 0.8$ and $\alpha = 1.2$, respectively. First of all, in both figures the trend is decreasing as the price per unit of storage increases. Furthermore, the α parameter determines the number of objects that are worth to be cached in the network. In particular, the higher the alpha, the lower their number: $\alpha = 1.2$ demands on average 60% less objects than $\alpha = 0.8$.

The price sensitivity with respect to the total traffic is instead depicted in Fig. 4.8c-4.8d for the Abilene topology, and in Fig. 4.8e-4.8f for Géant. While considering $\alpha = 0.8$ as in Fig. 4.8c and 4.8e we observe that even when the price ratio is very competitive (and equal to 10^{-2}), the randomized rounding heuristic only finds a suboptimal solution, whereas the greedy heuristic is very close to the optimum.

Another interesting observation is that when we consider a higher Zipf $\alpha = 1.2$ as in Fig. 4.8d and 4.8f, the effect of the different price ratio become instead negligible, for a very large span of values. This observation is remarkable, especially when considering



(a) *Abilene Installed Storage* $\alpha = 0.8$

(b) *Abilene Installed Storage* $\alpha = 1.2$

(c) *Abilene Overall Traffic* $\alpha = 0.8$

(d) *Abilene Overall Traffic* $\alpha = 1.2$

(e) *Géant Overall Traffic* $\alpha = 0.8$

(f) *Géant Overall Traffic* $\alpha = 1.2$

Figure 4.8: Price Plots. *In Fig. 4.8a-4.8f we show the sensitivity to different pricing policies for the migration to NDN. Fig. 4.8a-4.8b refer to the amount of storage installed in the Abilene topology, while in Fig. 4.8c-4.8f we represent the overall traffic costs for the Géant and Abilene topologies, as a function of the* storage vs. migration price ratio*, which is defined as* $\frac{C^S}{C^M}$*, where the* C^M *value is fixed to* $C^M = \max_p$.

the real benefits that a future deployment of this technology may achieve: for skewed popularity distributions, the pricing policy has only marginal effects.

4.5.6 Effect of Unsplittable Routing

Our proposed optimization model for the IP network enforces unsplittable routing conditions, whereas in the NDN formulation, only NDN-migrated routers can split flows on multiple paths. In this subsection we relax both these assumptions and consider the *splittable* scenario, according to which network nodes can route a single flow on multiple paths.

In Tables 4.2-4.3 we show the results we obtained comparing the performance of

Table 4.2: *Splittable and unsplittable routing, IP network, average values.*

Topology	Average Traffic Increase		Average Cost Decrease	
	$\alpha = 0.8$	$\alpha = 1.2$	$\alpha = 0.8$	$\alpha = 1.2$
Abilene	3%	7%	2%	$< 1\%$
Airtel	1.2%	3.4%	$< 1\%$	$< 1\%$
Claranet	1.5%	5%	$< 1\%$	$< 1\%$
Géant	4%	8%	1%	$< 1\%$
Netrail	4.5%	10%	2%	1%

Table 4.3: *Splittable and unsplittable routing, IP network, max values.*

Topology	Max Traffic Increase		Max Cost Decrease	
	$\alpha = 0.8$	$\alpha = 1.2$	$\alpha = 0.8$	$\alpha = 1.2$
Abilene	13.5%	20%	10.7%	10%
Airtel	19%	50%	6%	4.5%
Claranet	27.5%	28.6%	7%	9.6%
Géant	31%	77%	8%	6.8%
Netrail	28%	43%	8%	12%

splittable and unsplittable routing for IP networks. In particular, in Table 4.2 we show the average values, while in Table 4.3 we show the maximum results we obtained in terms of the traffic increase that a splittable IP network can accommodate, as well as the cost savings experienced.

The rationale behind the choice of these metrics is that, in the chosen instances of our network model, by letting intermediate nodes support splittable routing we increase the size of the feasibility region making the network forward up to 50% more traffic than the traffic it can deliver in the unsplittable scenario (Airtel topology, Table 4.3). On the other hand, in terms of cost benefits, splittable routing leads to very modest cost savings.

By comparing the results provided in Table 4.2 with those in Table 4.3 we can conclude that there exist a large gap between the values observed on average, and those experienced in the worst case scenarios. In particular, there are some instances in which splittable routing can significantly improve the performance, reducing costs up to 12% (Netrail topology, Table 4.3). However, the average cost benefits accountable to splittable routing are negligible, and lead to savings up to 2%, considering the average values (Table 4.2). Finally, we observe that most of the benefits can be obtained when the content popularity is more skewed, as in the case where the Zipf alpha exponent is larger.

We considered the impact of splittable routing also in our models for the NDN network, and we discovered that it has a negligible impact on the final solution, leading to benefits accountable to much less than 1% with respect to both the traffic increase and cost decrease metrics. This behavior is caused by the fact that the impact of content caching dramatically pushes the network content distribution capabilities to its boundary, making it extremely challenging to further improve the network performance.

4.6 Related Work

In this section we survey relevant literature on optimal cache placement and request routing for in-network content dissemination.

A pioneering work by Krishnan et al. is presented in [113], and deals with cache placement in a TCP/IP network to minimize the overall network flow. Among the key-features of their formulation, we mention that they bound the number of caches that can be installed, moreover they assume the average *flow hit-rate* is given as an input parameter. Wang et al. formulate in [185] a model to solve a storage constrained cache allocation problem with optimal object placement in NDN. They focus the analysis on discovering which parameters mostly affect the location of caches in the topology. In [90], Hasan et al. tackle the problem of minimizing the overall cost for inter-Autonomous System cache deployments in transit ISP networks, considering the server, energy and bandwidth prices. Finally, optimal content-oriented request routing is investigated by Mihara et al. in [132]. They minimize the overall traffic on the most congested link, however caching is not considered in their analytic framework.

In [44] Chai et al. present a cache decision policy for Information-Centric Networking based on the measure of the betweenness-centrality. Rather than supporting ubiquitous in-network caching, in their proposal objects tend to be cached in preferential (central) positions in the topology. In [67], Fayazbakhsh et al. claim that the benefits introduced in NDN by pervasive caching and nearest-replica routing can be obtained with an adequate CDN infrastructure. In particular this option can dramatically reduce the costs to migrate to NDN. However, rather than considering the best solution of the problems, the authors instead take into account fixed cache placement techniques as well as cache provisioning solutions and do not instead look for the best allocation possible.

Our MILP formulation differs from previous works for the following reasons: (1) we accurately model link capacities and traffic flows, (2) we explicitly take into account the contents (i.e, the objects), (3) we adopt an economic perspective on the subject, solving the network planning problem for the migration to an NDN and (4) we jointly solve the optimal *request routing*, *cache provisioning* and *object placement* problems in a budget-constrained scenario.

4.7 Conclusion

In this chapter we tackled the *content-aware network planning* problem for the *migration* to an NDN, in a *budget constrained* scenario. In order to derive the optimal strategy that the operator should pursue, we formulated a Mixed Integer Linear Programming model that can be used to jointly identify the node migration strategy, with optimal object placement and request routing. Our proposed optimization model takes into account economic parameters related to: (1) the traffic, (2) the router migration and (3) the caching storage costs. We further complemented our contribution by extending our previous works with the design of a novel greedy heuristic that can solve the planning problem cutting the computation time of an order of magnitude and find-

ing much better solutions than those computed with our previous randomized rounding heuristic.

We discovered that, by migrating only few nodes to NDN, the operator can experience up to a 68% reduction in traffic costs, compared to those of an IP network, as we observed for the Géant topology. On top of that, when the content popularity distribution is very skewed (i.e, $\alpha = 1.2$) the migrated nodes have on average 87% less storage than the one deployed when setting $\alpha = 0.8$. We also observed that the migration prices have a modest impact on the overall migration costs, and migrating to NDN leads to significant benefits for a large span of migration prices. Numerical results show that our proposed greedy heuristic can compute solutions practically overlapping the optimal one for the Abilene topology, and on average only 5% costlier than optimum in the Géant topology, while reducing the computation time to just few milliseconds even for the large topologies we took into account.

CHAPTER 5

Content Distribution Under Time-Varying Popularity

In order to cope with the huge amount of content-driven traffic demands, Content-Delivery Networks (CDNs) are currently used as a well-established technology to serve clients' requests through an infrastructure that is not specifically tailored for one such purpose. On the other hand, the novel paradigm of Named-Data Networking (NDN) aims at filling the gap of this misalignment by changing the network-layer protocols, solving the content-distribution problem at its root.

In this chapter, we use optimization models to analyze the performance gains that CDN and NDN can achieve, by reducing the total amount of traffic exchanged through the network. Rather than considering the long-term planning as done in Chapter 4, we tackle this problem by adopting a time-varying content popularity evolution model that takes into account the dynamic behavior of users.

We discover that, in most of the cases, CDN leads to better performance, reducing the amount of traffic the network should deliver, whereas NDN should instead be preferred in those scenarios where CDN cannot quickly react to popularity evolution. On top of that, we show that very limited benefits can be obtained by changing the cache replacement algorithms.

5.1 Introduction

By moving content replicas closer to the actual consumers location, Content-Delivery Networks (CDNs), such as Akamai [140], are currently used to efficiently serve the content requests of worldwide Internet users, in today's TCP/IP Internet. CDNs are also used

to effectively respond to sudden popularity changes, known as *flash crowds*, which can mine the reliability of the system by overwhelming the servers with a huge number of requests.

Moved by the desire (and necessity) to understand whether the migration towards NDN can provide significant benefits to network providers, in this chapter we analyze and compare the performance of the NDN and CDN architectures. We consider a scenario where *time-varying* content popularity demands are generated by the consumers, in such a way that we can assess the network capability to react to the dynamic content popularity evolution. We formulate novel optimization models to represent relevant features for the NDN and CDN architectures, and we use them to analyze the best performance bounds that these networks can achieve.

Our key findings suggest that when the content popularity evolves very quickly, NDN minimizes the overall network traffic, while CDN should instead be preferred whenever the content popularity dynamics evolves at a slower pace. Another take-home message of our work, in line with the results presented in other papers (i.e., [67]), is that it is better to deploy caching storage on a limited number of nodes rather than distributing it uniformly throughout the network.

Our main contributions can be summarized as follows:

1. We formulate a novel optimization model to study the performance bounds of a Named-Data Network, solving the joint object placement and routing problem, under a realistic, *time-varying* object popularity evolution scheme and with the most notable cache replacement policies [155].
2. We formulate an optimization model to represent a similar scenario in a Content-Delivery Network (CDN), where replica servers are distributed according to the *k-median* model [148]. We build the CDN model in a way such that we can control the speed of reaction of the network to content popularity changes.
3. We extensively discuss the obtained results, and show that in the considered scenarios, with the Netrail topology, NDN reduces the traffic more consistently than CDN only if this latter is at least 15 times slower to react to popularity changes. On the other hand, in the larger Géant topology, we observed that CDN is always to be preferred since it reaches up to 14% better performance than NDN.

This chapter is structured as follows: Sec. 5.2 motivates the choices we made to evaluate the content distribution performance of the network. Sec. 5.3 describes the content popularity evolution model. Sec. 5.4 illustrates the proposed optimization models used to study the performance bounds. Numerical results obtained solving these models are presented and discussed in Sec. 5.5. In Sec. 5.6 we discuss related works. Finally, concluding remarks are presented in Sec. 5.7.

5.2 Evaluating Content Distribution Performance

In this section, we describe the rationale behind the evaluation of the performance of NDN and CDN.

In particular, we would like to consider a fair scenario that does not introduce biases neither in favor nor against one specific network paradigm. We assume that the most realistic case is the one where the object popularity evolves in time, making the traffic demand profile be expressed as an input parameter that is *time-variant*. In our models, time is a discrete quantity expressed as a finite set of time-slots, denoted with $\mathcal{T}$ and such that each $t \in \mathcal{T}$ has a fixed duration. Moreover, since we would like to perform such evaluation under different conditions of content popularity evolution (slow/medium/fast evolution speed), we should adopt a popularity evolution model that depends on few parameters (ideally only one), and that well represents the burstiness of object traffic demands.

The performance metric that we take into account is the *total traffic* exchanged in the network, which should be minimized. The models then perform optimal *object placement* and *routing* choices for both the NDN and CDN architectures.

A given amount of caching storage, denoted with S, is uniformly distributed on all the caching routers $r \in R$ in the NDN model. On the other hand, in the CDN model, only CDN nodes have caching capabilities. We denote with $\mathcal{D}$ the set of CDN nodes, whose cardinality is restricted to be $N = |\mathcal{D}|$, where $N \leq |R|$. The same total caching storage is distributed in the two networks, but, due to their lower cardinality, each CDN node will usually store more objects than those persisted in a NDN node.

CDN nodes are pre-allocated optimally using a well-known *replica server placement* model. This assumption realistically models a CDN, since its owner will choose to deploy the machines only in the locations where they are mostly useful.

Another clear design requirement that our models meet is that there must by a tunable parameter that lets us change the speed at which the CDN can adapt to sudden popularity changes. We call this parameter "*Relocation Time*" ($\mathcal{T}_r$). In our CDN model, the relocation time is a subset of consecutive time-slots $\mathcal{T}_r \subseteq \mathcal{T}$, and it is used to represent a time window under which CDN nodes cannot change the objects they persist in their storage. By setting $|\mathcal{T}_r| = 1$, we are forcing object relocation in CDN to have the same dynamics of the popularity evolution model. Since we can say that NDN can always promptly react to popularity changes, with $|\mathcal{T}_r| = 1$ CDN is "*as reactive as*" NDN; instead, when we set $|\mathcal{T}_r| = 10$, for instance, it means that CDN is 10 times slower than NDN.

A relevant feature that our models take into account is the fact that a CDN is run by a single owner in a centralized manner: at a given point in time the owner can choose to send a replica of a given object on any node in the network, by pushing it towards that destination, for instance because it performed popularity forecasting and chose to pre-fetch an object on a given server. The same condition does not apply to the NDN model, where instead we restrict the objects that a given router can cache to the subset of those it forwarded in the past. In the numerical results we explicitly study the cost to move the objects to the CDN surrogates, and we show that it has negligible impact on the considered performance metric.

Due to their inner differences, we strongly support the idea that NDN and CDN should complement each other rather than being perceived as two antagonist models. In particular, our numerical results give evidence that NDN should be preferred whenever the content popularity dynamics evolves very quickly, while in the other cases CDN

Algorithm 9: Popularity Evolution, Rank-shift Model

Input : r_o^t, O, ρ
Output: r_o^{t+1}

```
for o ∈ O do
    if UniformRandom(0,1) ≤ ρ then
        r' ⇐ UniformDiscreteRandom(1, r_o^t);
        for o' ∈ O do
            if r' ≤ r_{o'}^t ≤ r_o^t then
                r_{o'}^{t+1} ⇐ r_{o'}^t + 1;
            end
        end
        r_o^{t+1} ⇐ r';
    end
end
```

leads to the lower overall traffic exchanged in the network. At the same time, we recognize that the simplicity of NDN should reduce the management costs with respect to CDN, but such type of analysis is out of the scope of our work.

5.3 Content Popularity Evolution Model

This section discusses the content popularity evolution model we used to generate synthetic traffic demand traces. Many research papers (e.g., [27, 68, 150]) agree in supporting the idea that the content popularity dynamics is highly non-stationary, and characterized by a bursty and oscillatory behavior, mostly governed by exogenous events. In this subsection we will accurately describe the synthetic traffic model, based on the proposal of Ratkiewicz et al. [150], that we use to generate the input traffic demand mimicking non-linear popularity shifts for a fixed-size object catalog denoted with O. Despite the fact that content churn [1] may increase the realism, it is a common assumption made in the CDN and NDN literature to consider a fixed-size content catalog [39, 142].

Time is discretized into a finite set of time-slots, denoted with $\mathcal{T}$. Each object $o \in O$ has a time-dependent rank parameter r_o^t which describes how likely that object will be requested during $t \in \mathcal{T}$. The lower the rank of an object, the higher the likelihood that such content will receive requests in that time-slot. We assume all the time slots $t \in \mathcal{T}$ last for the same (and constant) amount of time and such that the popularity rank of every object in t does not change.

Given the object rank r_o^t, the Zipf discrete distribution is often used in the literature to represent the popularity of Internet contents, since it was shown that it is an adequate model for it [30, 67, 142]. The Zipf distribution is characterized by the popularity exponent α: the higher the α, the more skewed the requests are.

1. The churn is a measure of the number of individuals moving in or out a given collective over a specific period of time.

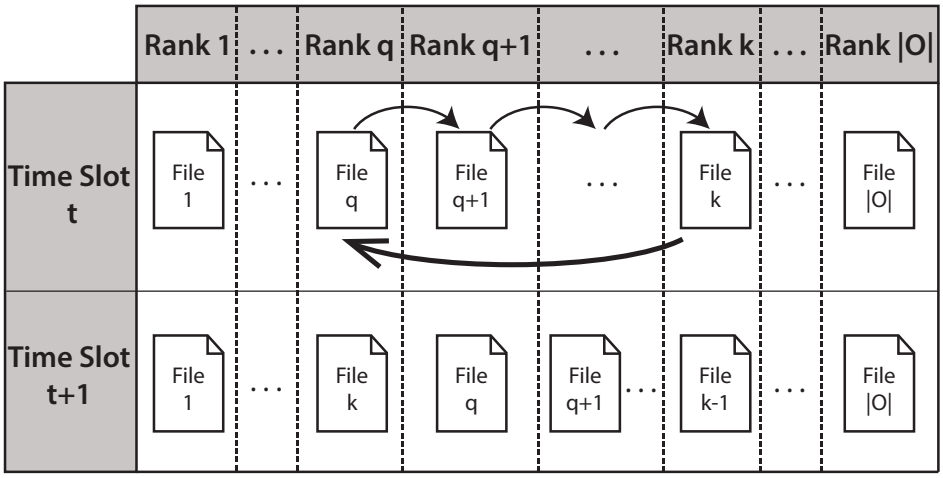


Figure 5.1: The Rank-Shift Model. *In this example, at time slot $t+1$, the popularity of the k-th ranked file is updated to the value of q (randomly chosen), thus all the files from q to $k-1$ are shifted of one rank.*

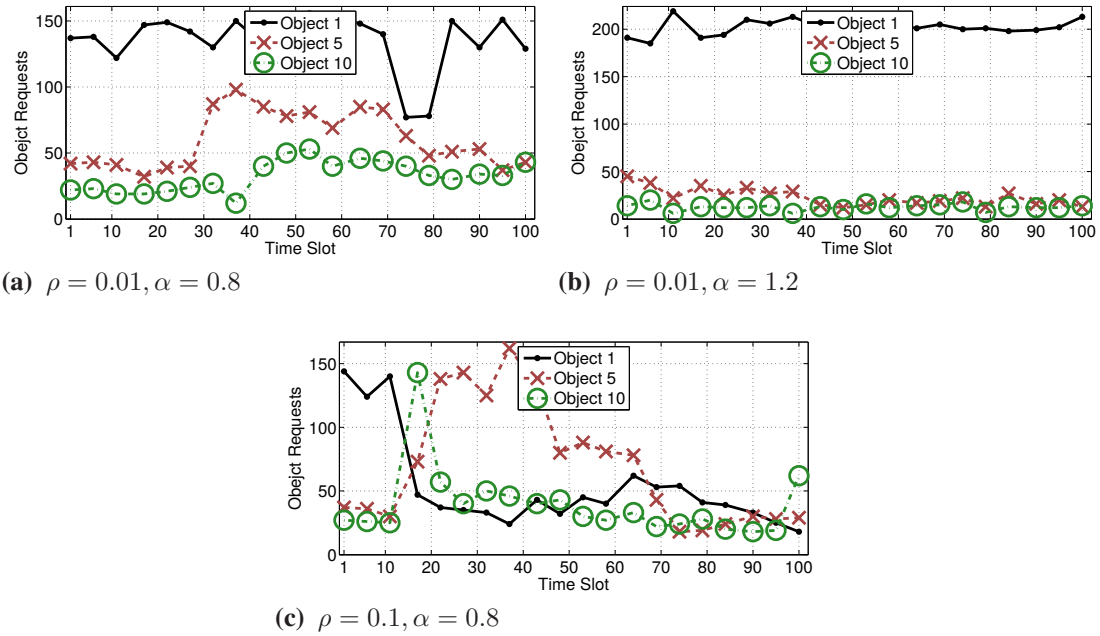
The time-dependent popularity dynamics is governed by the rank evolution parameter ρ in the 0 to 1 range. At each time slot, the ranking of a given object might evolve to become more popular in the immediate future. We control the speed at which this change happens through the usage of ρ: the higher the ρ value, the faster the popularity evolves. In the extreme cases, by setting $\rho = 0$ we neglect popularity evolution, whereas $\rho = 1$ removes temporal correlation of the requests. When the rank r_o^t of an object is updated, it makes the other objects' rank be shifted to a new (less popular) value.

Algorithm 9 illustrates the pseudocode used to compute the future object rank, given its current value. For each object (Step 1), with probability ρ (Step 2), the algorithm randomly selects a lower popularity class r' (Step 3), making the object suddenly become more popular. In Step. 4, the rank of the other objects is instead shifted to a reduced popularity level, in order to make sure that the popularity rank will be unique among all the objects. An example of object popularity evolution is shown in Fig. 5.1, where at time slot $t+1$, the popularity of the k-th ranked file is updated to the new value of q, randomly chosen.

As illustrated in Figures 5.2a-5.2c, by changing the ρ and α parameters, we can easily mimic very different behaviors: a range of different content requests can be generated with such model, thus allowing us to well represent almost any type of request generation process. In particular, by decreasing the α value, content requests are less polarized towards the most popular objects, while higher ρ values make the content popularity dynamics evolve faster.

5.4 Network Models for Content Distribution

This section discusses the proposed optimization models that we use to study the performance of the NDN and CDN paradigms. Sec. 5.4.1 presents the *Object Routing*



(a) $\rho = 0.01, \alpha = 0.8$ **(b)** $\rho = 0.01, \alpha = 1.2$

(c) $\rho = 0.1, \alpha = 0.8$

Figure 5.2: Time-Dependent Object Request Evolution. *The figures plot the time-dependent object request evolution as a function of the Zipf α exponent and the rank evolution probability ρ, considering 100 time slots and 10 objects. The figures show the evolution of requests for the objects that in the first time slot have rank 1, 5 and 10, being 1 and 10 the most and least popular ranks, respectively.*

model (OR) with *time-varying* demands. In Sec. 5.4.2 and Sec. 5.4.3, we tailor the OR model to better represent relevant characteristics of NDN and CDN, respectively.

5.4.1 Object Routing Model with Time-Varying Demands

In this subsection we describe our proposed *Object Routing model* (OR) with *time-varying* demands. Such model well describes a TCP/IP network that does not have any caching functionality. In the following subsections we will further extend the OR model, taking into account relevant features that characterize NDN and CDN architectures. As summarized in Table 5.1, three extensions of the OR model will be presented:

1. *Object Allocation and Routing (OAR)*, a model tailored for NDN, that finds the optimal solution on the overall time horizon;
2. *OAR - Single Time Slot Heuristic (OAR-TS)*, tailored for NDN, which chooses the optimal solution for single time slots;
3. *OAR - Single Relocation Time CDN Heuristic (OAR-TR-CDN)*, which is the equivalent for CDN of OAR-TS.

We model the network as an undirected graph $G(N, E)$, where N is the set of nodes and E the set of edges. The set of nodes N is partitioned into three disjoint sub-sets: *consumers* (denoted by C), *producers* (denoted by P) and *routers* (denoted by R), such

Table 5.1: *Summary of the Network Optimization Models we propose in this chapter.*

Model Name	Caching Strategy	Time Horizon	Network Type	Section
Object Routing (OR)	-	Many Slots	TCP/IP	5.4.1
Object Allocation and Routing (OAR)	Optimal	Many Slots	NDN	5.4.2
OAR - Single Time Slot Heuristic (OAR-TS)	Optimal LFU Random	Single Slot	NDN	5.4.2
OAR - Single Relocation Time CDN Heuristic (OAR-TR-CDN)	Optimal	Single Slot	CDN	5.4.3

that $V = C \cup P \cup R$. Furthermore, we denote with $\mathcal{T}$ the set of time slots, whereas O represents the set of objects that can be retrieved from the network, also known as the *catalog*. It is important to mention that, as frequently assumed in the network planning literature, both the *consumer* and *producer* nodes must be interpreted as *test points* at which an aggregate of traffic demands can either be requested or served.

We denote with d_c^{to} the demand of consumer $c \in C$ for object $o \in O$ at time $t \in \mathcal{T}$. The demand is expressed in data size units (e.g., Mbytes); moreover, we assume that traffic requests are *inelastic*, meaning that they cannot be postponed to subsequent time slots. In order to satisfy the demands, producers can serve the subset of objects they own. For each producer $p \in P$ and object $o \in O$, we denote with a_p^o the producer-object allocation, where:

$$a_p^o = \begin{cases} 1, & \text{if object } o \text{ is available at producer } p \\ 0, & \text{otherwise.} \end{cases} \tag{5.1}$$

All the routers in the network perform routing and forwarding. Considering the producer-router pairs $(p, r) \in P \times R$, we denote the corresponding link capacity with b_{pr}. Similarly, b_{rc} and $b_{r_1 r_2}$ denote the router-consumer $(r, c) \in R \times C$, and the router-router $(r_1, r_2) \in R \times R$ link capacity, respectively.

We denote with y_{pr}^{to} the producer-router time-dependent flow variable which represents the amount of data that producer $p \in P$ sends to router $r \in R$ for object $o \in O$ in time slot $t \in \mathcal{T}$. Similarly, we define the router-consumer flow variable as y_{rc}^{to}, and the router-router flow variable with $y_{r_1 r_2}^{to}$. For the sake of clarity, Table 5.2 summarizes the notation we use in this chapter.

We begin by describing the *Object Routing* model we use to compute the optimal packet routing to minimize the overall network traffic. By solving this model, we determine the performance bound of a network that does not support any type of caching functionality. In the following sections (viz., Sec. 5.4.2 and 5.4.3) we will extend the OR model in order to describe the behavior of a NDN and a CDN architecture, respectively. Given the above definitions and assumptions, we formulate the optimal

Table 5.2: *Summary of the notation used in our optimization models*

Parameters	
C	Set of Consumers test points
P	Set of Producers test points
R	Set of Routers
O	Set of Objects
$\mathcal{T}$	Set of Time Slots
$\mathcal{D}$	Set of CDN Nodes test points
d_c^{to}	Traffic demand generated by $c \in C$ for $o \in O$ in $t \in \mathcal{T}$
b_{rc}	Link capacity between $r \in R$ and $c \in C$
b_{pr}	Link capacity between $p \in P$ and router $r \in R$
$b_{r_1r_2}$	Link capacity between router $r_1 \in R$ and router $r_2 \in R$
a_p^o	0-1 parameter to indicate if producer $p \in P$ is publishing object $o \in O$
S	Maximum number of objects that each router can cache
Q	A large number
N	Maximum number of CDN nodes that can be deployed
m	Maximum memory that a CDN node can use for caching
M	Total memory shared by all CDN nodes for caching

Decision Variables	
x_r^{to}	0-1 variable indicating if router $r \in R$ is caching $o \in O$ at time $t \in \mathcal{T}$
$y_{r_1r_2}^{to}$	Flow of $o \in O$ from $r_1 \in R$ to $r_2 \in R$ during $t \in \mathcal{T}$
y_{pr}^{to}	Flow of $o \in O$ from $p \in P$ to $r \in R$ during $t \in \mathcal{T}$
y_{rc}^{to}	Flow of $o \in O$ from $r \in R$ to $c \in C$ during $t \in \mathcal{T}$

Object Routing model (OR) with *time-varying* demands as follows:

$$\min \sum_{\substack{\forall o \in O \\ \forall t \in \mathcal{T}}} \left(\sum_{\substack{\forall r_1 \in R \\ \forall r_2 \in R}} y_{r_1r_2}^{to} + \sum_{\substack{\forall p \in P \\ \forall r \in R}} y_{pr}^{to} + \sum_{\substack{\forall r \in R \\ \forall c \in C}} y_{rc}^{to} \right) \tag{5.2}$$

subject to:

$$\sum_{\forall c \in C} y_{r_1c}^{to} + \sum_{\forall r_2 \in R} y_{r_1r_2}^{to} = \sum_{\forall p \in P} y_{pr_1}^{to} + \sum_{\forall r_2 \in R} y_{r_2r_1}^{to} \qquad \forall (r_1, o, t) \in R \times O \times \mathcal{T} \tag{5.3}$$

$$\sum_{\forall o \in O} y^{to}_{r_1 r_2} \leq b_{r_1 r_2} \qquad \forall (r_1, r_2, t) \in R \times R \times \mathcal{T} \quad (5.4)$$

$$\sum_{\forall o \in O} y^{to}_{pr} \leq b_{pr} \qquad \forall (p, r, t) \in P \times R \times \mathcal{T} \quad (5.5)$$

$$\sum_{\forall o \in O} y^{to}_{rc} \leq b_{rc} \qquad \forall (c, r, t) \in C \times R \times \mathcal{T} \quad (5.6)$$

$$\sum_{\forall r \in R} y^{to}_{rc} = d^{to}_{c} \qquad \forall (c, o, t) \in C \times O \times \mathcal{T} \quad (5.7)$$

$$y^{to}_{pr} \leq b_{pr} \cdot a^{o}_{p} \qquad \forall (p, r, o, t) \in P \times R \times O \times \mathcal{T} \quad (5.8)$$

$$y^{to}_{r_1 r_2} \geq 0 \qquad \forall (r_1, r_2, o, t) \in R \times R \times O \times \mathcal{T} \quad (5.9)$$

$$y^{to}_{pr} \geq 0 \qquad \forall (p, r, o, t) \in P \times R \times O \times \mathcal{T} \quad (5.10)$$

$$y^{to}_{rc} \geq 0 \qquad \forall (r, c, o, t) \in R \times C \times O \times \mathcal{T}. \quad (5.11)$$

The objective function (5.2) minimizes the total traffic transferred across all network links.

The set of constraints (5.3) imposes the flow balance condition at each router. Constraints (5.4), (5.5) and (5.6) bound the total link capacity that can be used on router-router, producer-router and router-consumer links, respectively. The set of constraints (5.7) makes sure that the network satisfies, in each time slot, all consumers' demand. Producers can only serve the subset of objects they possess, and this condition is expressed by the set of constraints (5.8). Lastly, constraints (5.9), (5.10) and (5.11) impose that router-router, producer-router and router-consumer flows are non-negative.

As represented in the OR model, the behavior of the network in each time slot is such that it does not depend on the solution obtained in other time slots. Therefore, it is possible to speed-up the model resolution by solving the routing problem independently in each time slot, and then aggregating the solution to compute the final objective function value.

5.4.2 Model for Content-Centric Network

In this section, we present the *Object Allocation and Routing* (OAR) model, an extension of the OR model tailored for NDN.

The OAR model extends the OR model adding *caching capabilities* to the routers. Given router $r \in R$, object $o \in O$ and time slot $t \in \mathcal{T}$, the router cache state is denoted by the binary variable x^{to}_{r}, which is such that:

$$x^{to}_{r} = \begin{cases} 1, & \text{if object } o \text{ is cached at router } r \text{ at time slot } t \\ 0, & \text{otherwise.} \end{cases} \quad (5.12)$$

In particular, we model a network where the cache is uniformly spread among all the nodes. Our model solves the *joint* optimal Object Allocation and Routing (OAR) problem, where the consumers express a *time-varying demand*. We can therefore formulate the mixed integer linear programming (MILP) model for OAR as follows:

$$\min \sum_{\substack{\forall o \in O \\ \forall t \in \mathcal{T}}} \left(\sum_{\substack{\forall r_1 \in R \\ \forall r_2 \in R}} y^{to}_{r_1 r_2} + \sum_{\substack{\forall p \in P \\ \forall r \in R}} y^{to}_{pr} + \sum_{\substack{\forall r \in R \\ \forall c \in C}} y^{to}_{rc} \right) \tag{5.13}$$

subject to constraints (5.4)-(5.11), and:

$$\sum_{c \in C} y^{to}_{r_1 c} + \sum_{r_2 \in R} y^{to}_{r_1 r_2} \leq Q \cdot x^{to}_{r_1} + \sum_{r_2 \in R} y^{to}_{r_2 r_1} + \sum_{p \in P} y^{to}_{p r_1} \qquad \forall (r_1, o, t) \in R \times O \times \mathcal{T} \tag{5.14}$$

$$x^{t,o}_r \leq x^{t-1,o}_r + \sum_{\forall r_2 \in R} y^{t-1,o}_{r_2 r} + \sum_{\forall p \in P} y^{t-1,o}_{pr} \qquad \forall (r, o, t) \in R \times O \times (\mathcal{T} \setminus \{0\}) \tag{5.15}$$

$$\sum_{\forall o \in O} x^{to}_r \leq S \qquad \forall (r, t) \in R \times \mathcal{T} \tag{5.16}$$

$$x^{0,o}_r = 0 \qquad \forall (r, o) \in R \times O \tag{5.17}$$

$$x^{to}_r \in \{0, 1\} \qquad \forall (r, o, t) \in R \times O \times \mathcal{T}. \tag{5.18}$$

The objective function (5.13) is the same as the one proposed for the OR model. The set of constraints (5.14) imposes flow-balance at each router, by also taking into account its caching capabilities. In particular, Q is a large number such that, when router r_1 is caching object o at time t (that is, $x^{t,o}_{r_1} = 1$), r_1 can directly serve all the incoming requests for that object. Link capacity constraints imposed by (5.4)-(5.6) are still valid in OAR, even when the router behaves as a cache.

In NDN, each node acts independently from all the others by choosing which content it should store according to the *local* information available. This means that each node can choose to store a content in its local cache if and only if in the previous time slot it has forwarded such object, or if it was already caching such data. This constraint is imposed by (5.15) on all the caching routers in the network.

In (5.16), we make sure that each caching router stores at most S objects in its local cache. In (5.17), we make sure that the caches are empty at the initial time slot (in a sort of initialization step); finally, the set of constraints (5.18) forces the variable x^{to}_r to be binary. It is important to note that the OAR model supports multipath packet routing, a native feature provided by NDN, and it also supports *off-path* caching. In fact, each router chooses the face on which packets should be forwarded knowing also the availability of content replicas in the caching storage of all the other nodes. Supporting off-path caching is in line with our goal of computing the performance bound of the NDN network.

Due to the fact that NDN routers can cache only the subset of objects they have seen in the immediate past, current network behavior strongly influences possible future choices, making every time slot be linked to all the others, as modeled by constraints (5.15). However, finding the optimal solution over the complete set of time slots is computationally very expensive, since the model has to consider all the demands on a global scale. Moreover, this optimization strategy assumes that the model can perform the optimal choice by knowing also *"future"* demands.

As shown by numerical results presented in Sec. 5.5.1, knowing the future (as OAR does) seems to provide only negligible improvements to the objective function, when compared to the alternative case where the solver knows the current demands and the previous state of the system. We call this latter model OAR *Single Time Slot Heuristic* (OAR-TS). OAR-TS lets us find a close to optimal solution to OAR, saving significant amount of time for the computation. The OAR-TS model is illustrated hereafter:

$$\min \sum_{\forall o \in O} \left(\sum_{\substack{\forall r_1 \in R \\ \forall r_2 \in R}} y^o_{r_1 r_2} + \sum_{\substack{\forall p \in P \\ \forall r \in R}} y^o_{pr} + \sum_{\substack{\forall r \in R \\ \forall c \in C}} y^o_{rc} \right) \tag{5.19}$$

subject to:

$$\sum_{c \in C} y^o_{r_1 c} + \sum_{r_2 \in R} y^o_{r_1 r_2} \leq Q \cdot x^o_{r_1} + \sum_{r_2 \in R} y^o_{r_2 r_1} + \sum_{p \in P} y^o_{p r_1} \qquad \forall (r_1, o) \in R \times O \tag{5.20}$$

$$\sum_{\forall o \in O} y^o_{r_1 r_2} \leq b_{r_1 r_2} \qquad \forall (r_1, r_2) \in R \times R \tag{5.21}$$

$$\sum_{\forall o \in O} y^o_{pr} \leq b_{pr} \qquad \forall (p, r) \in P \times R \tag{5.22}$$

$$\sum_{\forall o \in O} y^o_{rc} \leq b_{rc} \qquad \forall (c, r) \in C \times R \tag{5.23}$$

$$\sum_{\forall r \in R} y^o_{rc} = d_c \qquad \forall (c, o) \in C \times O \tag{5.24}$$

$$y^o_{pr} \leq b_{pr} \cdot a^o_p \qquad \forall (p, r, o) \in P \times R \times O \tag{5.25}$$

$$\sum_{\forall o \in O} x^o_r \leq S \qquad \forall r \in R \tag{5.26}$$

$$x^o_r \leq K^o_r \qquad \forall (o, r) \in O \times R \tag{5.27}$$

$$y^o_{r_1 r_2} \geq 0 \qquad \forall (r_1, r_2, o) \in R \times R \times O \tag{5.28}$$

$$y^o_{pr} \geq 0 \qquad \forall (p, r, o) \in P \times R \times O \tag{5.29}$$

$$y^o_{rc} \geq 0 \qquad \forall (r, c, o) \in R \times C \times O \tag{5.30}$$

$$x^o_r \in \{0, 1\} \qquad \forall (r, o) \in R \times O. \tag{5.31}$$

The objective function (5.19) as well as constraints (5.20)-(5.26) and (5.28)-(5.31) can easily be derived from the corresponding constraints of the OAR formulation, by removing the time-slot index.

We model time-slots relations in constraints (5.27), where in each time slot $t \in \mathcal{T}$ we use the binary parameter K^o_r to impose restrictions on the subset of objects that a given router $r \in R$ can cache. In particular, we emulate the original behavior studied in the OAR formulation by setting $K^o_r = 0$ for the initial slot $t = 1$, whereas for the other time slots $t > 1$, we set $K^o_r = 1$ if, in $t - 1$, $x^o_r = 1 \vee \sum_{\forall r_1 \in R} y^o_{r_1 r} > 1$, otherwise we set $K^o_r = 0$. Such condition checks whether in the previous time slot the router was already caching object o, or it forwarded at least one traffic unit for it.

By using other strategies to set the K_r^o parameter we can find tighter performance bounds for *real caching policies*; in particular we can emulate the "*Least Frequently Used*" (LFU) policy, forcing caches to store the objects that are requested more frequently, as well as the "Random" caching policy, making caches store the objects randomly [19].

More in depth, we emulate the LFU policy by computing Ω_t^{ro}, the cumulative traffic that router $r \in R$ has forwarded for a given object $o \in O$ up to time slot $t \in \mathcal{T}$, which is defined as: $\Omega_t^{ro} = \Omega_{t-1}^{ro} + \sum_{r_2 \in R} y_{rr_2}^o + \sum_{c \in C} y_{rc}^o$. For each router, we sort the Ω_t^{ro} values in decreasing order; we then remove all those values referred to objects $o \in O$ that the router has not seen in the previous time slot $t-1$, i.e. all those objects such that the condition $x_r^o = 1 \vee \sum_{\forall r_1 \in R} y_{r_1 r}^o > 1$ does not hold. Let Ω_{St}^r be the S-th element of such sorted list. We set $K_r^o = 1$ if $\Omega_t^{ro} \geq \Omega_{St}^r$, otherwise it is set to zero. As a result, in $t+1$ the S most frequently requested objects are stored in the cache of the r-th router, according to its local cumulative traffic data.

The random cache policy can be implemented using even an easier algorithm: we randomly sample S objects such that for each of them, $o \in O$, the following condition holds: $x_r^o = 1 \vee \sum_{\forall r_1 \in R} y_{r_1 r}^o > 1$.

5.4.3 Model for Content-Delivery Network

In this section, we illustrate the model used to find the performance bound of a CDN architecture. Since we are mostly interested in studying the performance of the network *after* the physical deployment of the nodes, we assume that the replica server placement problem has already been solved *a-priori*. To solve the placement problem we use the *k-median* model [148] since it is a simple strategy that only depends on the nodes distribution in the network, and is frequently used in practice in the CDN planning.

An example showing how the *k-median* model distributes the CDN nodes in the Géant network topology [7] is shown in Fig. 5.3. In such example, 50 consumers and 5 producers are spread uniformly in the network that is composed of 40 routers, while the *k-median* solution distributes 5 CDN nodes placing them in positions close to the consumers' locations.

Once that we have found the optimal CDN replica server allocation, we can then solve the *joint object placement and request routing problem*, as we did in the previous section with the OAR and the OAR-TS model. Two important features that our optimization model accurately takes into account are:

(1) Each CDN node can cache whatever content available. In fact, the CDN owner can optimize such allocation pushing whatever content on whatever replica server.

(2) Since this content migration is quite costly, the objects stored in a CDN node evolve with slower frequency than the one that we can get for NDN.

It is straightforward to address requirement (1), since we can simply relax a constraint in the OAR-TS model formulation. On the other hand, in order to enforce requirement (2), we introduce another concept that we call "*Relocation Time*", $\mathcal{T}_r$. The

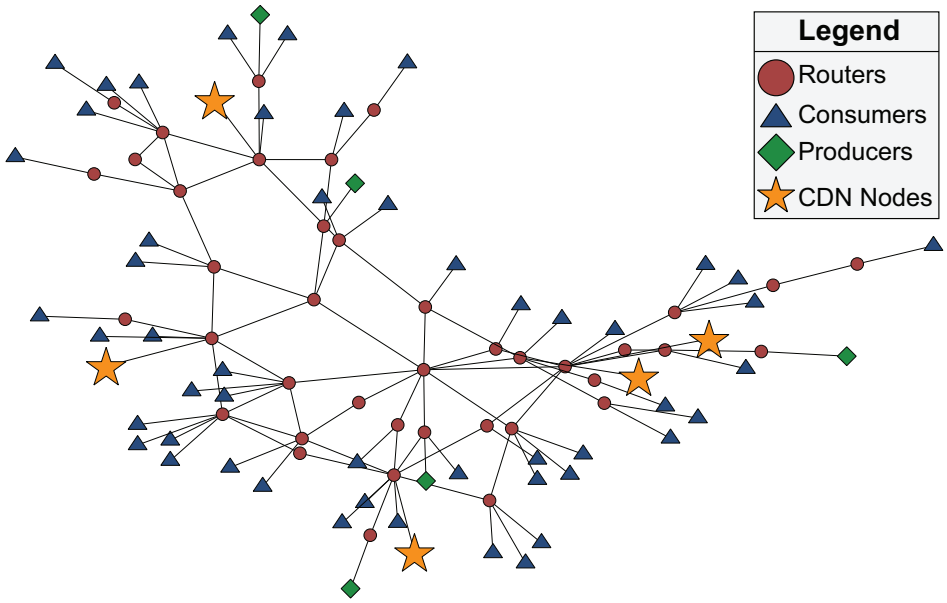


Figure 5.3: K-median model. *Example illustrating the placement of 5 CDN replica servers according to the solution of the* k-median model*, in the Géant network topology, with uniform distribution of 50 consumers and 5 producers.*

relocation time is a subset of consecutive time-slots $\mathcal{T}_r \subseteq \mathcal{T}$ during which the content cached in each CDN node is "frozen" and cannot be changed.

The set of CDN nodes is denoted with $\mathcal{D}$. We then formulate the *Optimal Allocation and Routing, Single Relocation Time Heuristic* model for CDN (OAR-TR-CDN) as follows:

$$\min \sum_{\forall o \in O} \left(\sum_{\substack{\forall r_1 \in R \\ \forall r_2 \in R}} y^o_{r_1 r_2} + \sum_{\substack{\forall p \in P \\ \forall r \in R}} y^o_{pr} + \sum_{\substack{\forall r \in R \\ \forall c \in C}} y^o_{rc} \right) \tag{5.32}$$

subject to:

$$\sum_{\forall c \in C} y^o_{r_1 c} + \sum_{\forall r_2 \in R} y^o_{r_1 r_2} = \sum_{\forall d \in \mathcal{D}} y^o_{d r_1} + \sum_{\forall p \in P} y^o_{p r_1} + \sum_{\forall r_2 \in R} y^o_{r_2 r_1} \quad \forall (r_1, o) \in R \times O \tag{5.33}$$

$$\sum_{\forall o \in O} y^o_{r_1 r_2} \leq b_{r_1 r_2} \cdot |\mathcal{T}_r| \qquad \forall (r_1, r_2) \in R \times R \tag{5.34}$$

$$\sum_{\forall o \in O} y^o_{pr} \leq b_{pr} \cdot |\mathcal{T}_r| \qquad \forall (p, r) \in P \times R \tag{5.35}$$

$$\sum_{\forall o \in O} y^o_{rc} \leq b_{rc} \cdot |\mathcal{T}_r| \qquad \forall (c,r) \in C \times R \quad (5.36)$$

$$\sum_{\forall o \in O} y^o_{dr} \leq b_{dr} \cdot |\mathcal{T}_r| \qquad \forall (d,r) \in \mathcal{D} \times R \quad (5.37)$$

$$\sum_{\forall r \in R} y^o_{rc} = \sum_{\forall t \in \mathcal{T}_r} d^{to}_c \qquad \forall (c,o) \in C \times O \quad (5.38)$$

$$y^o_{pr} \leq a^o_p \cdot b_{pr} \cdot |\mathcal{T}_r| \qquad \forall (p,r,o) \in P \times R \times O \quad (5.39)$$

$$\sum_{\forall o \in O} x^o_d \leq S' \qquad \forall d \in \mathcal{D} \quad (5.40)$$

$$y^o_{dr} \leq b_{dr} \cdot |\mathcal{T}_r| \cdot x^o_d \qquad \forall (d,r,o) \in \mathcal{D} \times R \times O \quad (5.41)$$

$$y^o_{r_1 r_2} \geq 0 \qquad \forall (r_1, r_2, o) \in R \times R \times O \quad (5.42)$$

$$y^o_{pr} \geq 0 \qquad \forall (p,r,o) \in P \times R \times O \quad (5.43)$$

$$y^o_{rc} \geq 0 \qquad \forall (r,c,o) \in R \times C \times O \quad (5.44)$$

$$y^o_{dr} \geq 0 \qquad \forall (r,d,o) \in R \times \mathcal{D} \times O \quad (5.45)$$

$$x^o_r \in \{0,1\} \qquad \forall (r,o) \in R \times O. \quad (5.46)$$

The OAR-TR-CDN objective function (5.32) is similar to the one proposed for the other models, since we want to minimize the overall network traffic.

In (5.33) we set the flow balance constraints, while in (5.34), (5.35), (5.36) and (5.37) we set the capacity constraints on router-router, producer-router, router-consumer and CDN-router flows, respectively. The overall consumer demand should be satisfied, as enforced by constraints (5.38), where we aggregate consumers' demands on all the time-slots in the relocation time $t \in \mathcal{T}_r$. In (5.39) we force producers to offer the subset of objects they own.

The set of constraints (5.40) limits the available caching space at each CDN router, while in (5.41) we limit the subset of objects that a CDN node can serve to those that it is caching. Finally, non-negativity constraints for flow variables are imposed in (5.42)-(5.45), while in (5.46), we make sure that the x^o_r variable is binary.

The relocation time is a subset of consecutive time slots, and is used in the OAR-TR-CDN model to aggregate the entire demand in one *virtual* slot. For this reason in constraints (5.34)-(5.37), (5.39) and (5.41) we multiply the capacity by $|\mathcal{T}_r|$, a non-dimensional quantity that is the number of time-slots considered in the given relocation time. Such choice permits to reduce the computational time required to solve the optimization problem.

Comments

The OAR-TS and OAR-TR-CDN models capture relevant characteristics of the NDN and CDN architectures, respectively. In particular, while in OAR-TS all the routers are equipped with caching storage, in OAR-TR-CDN only the subset of CDN nodes implement caching functionalities. Moreover, OAR-TS sets a constraint on the objects that each router can cache, restricting such subset to all the data that the router has forwarded in the previous time slot. On the other hand, CDN nodes can cache what-

Table 5.3: *Parameters of the Numerical Results*

Numerical Results Parameters	
Topologies	Netrail (7 nodes), Abilene (11 nodes) Sprint (11 nodes), Claranet (15 nodes) Airtel (16 nodes), Géant (40 nodes)
Number of Consumers	10 000, connected at 10 test-points
Number of Producers	5 000, connected at 5 test-points
Zipf α Exponent	$\{0.8, 1.2\}$
Content Popularity Evolution ρ	In the range from 10^{-4} to 0.99
Link Rate	50 000 objects per time slot
Consumer Demand	10 objects per time slot, per consumer
Cache Size per Router	In the range 1-5% of the total catalog
Catalog Size	10^7 objects - 100 popularity classes
Number of Time Slots	100
CDN Relocation Time	In the range 1-15, default value: 3
Number of CDN nodes	In the range 1-10, default value: 3

ever content they want, and we use the relocation time to control the speed at which the CDN reacts to popularity evolution. Furthermore, our analysis focuses on the performance bounds of NDN and CDN: in particular, we are not considering management cost savings that the NDN architecture can obtain with respect to CDN.

5.5 Numerical Results

This section presents the results obtained formulating the proposed optimization models in OPL and solving them using the CPLEX solver [176]. In Sec. 5.5.1 we compare the objective function obtained using OAR with the heuristic OAR-TS solution. Sec. 5.5.2 extensively presents and discusses the numerical results we recorded while comparing the performance bounds of NDN and CDN.

5.5.1 Comparison of OAR and OAR-TS

In this subsection we compare the results obtained using OAR and OAR-TS and show that, in the considered scenarios, the bounds of the two models are very close to each other, even though OAR is computationally more demanding than OAR-TS. Unless stated otherwise, Table 5.3 summarizes the parameters used to perform the analysis.

The topology we consider for this analysis is the Netrail topology with 10^7 objects partitioned into 100 popularity classes. We compute the optimal solution considering 100 time slots, while we attach consumers and producers to 10 and 5 test points, respectively. We run the OAR model limiting the overall running time to 10 hours and observing a final MIP gap always below 2% of the optimum solution. Traffic demands are generated setting $\alpha = 0.8, \rho = 0.2$. For the same scenario, CPLEX can always find

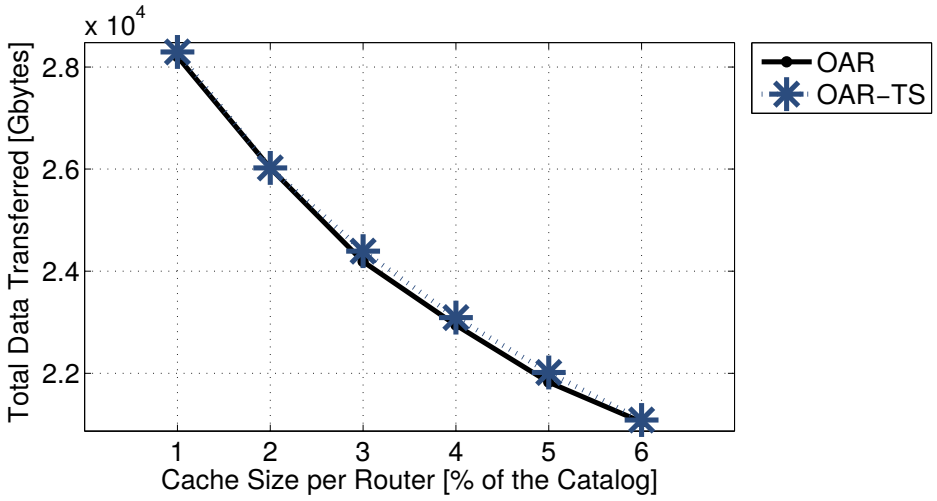


Figure 5.4: Comparison of OAR and OAR-TS. *We solve the Netrail topology with a catalog of 100 object classes and 100 time slots, with* $\alpha = 0.8, \rho = 0.2$. *The gap between the solution of OAR and OAR-TS is negligible (less than* 1%*).*

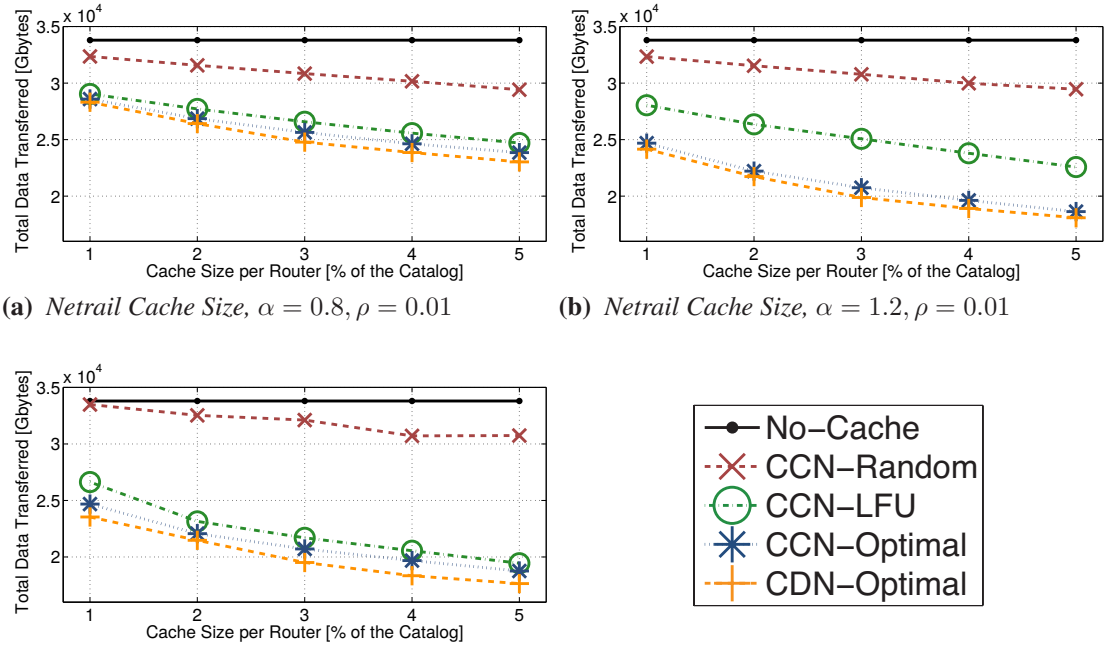
the optimal solution of the OAR-TS model in less than ten minutes for the whole time horizon.

As portrayed in Fig. 5.4, the OAR and OAR-TS curves are practically overlapping, in fact the gap between the two is always below 1%. This behavior clearly shows that knowing future demands (as OAR does) can lead only to negligible improvements in the objective function, and therefore it legitimates the use of our proposed OAR-TS heuristic to derive a close to optimal solution.

5.5.2 Performance Comparison of NDN and CDN

In this section we present the results obtained performing an extensive evaluation of the optimization models we designed. As shown in Table 5.3, we considered 6 different network topologies whose size is in line with [19, 120, 155, 156]. However, for the sake of brevity, hereafter we present the results obtained with Netrail and Géant, since the trends observed for the other topologies are in between the values observed for them.

As shown in Table 5.3, in all the scenarios we consider 10 000 consumers and 5 000 producers connected respectively to 10 and 5 test points spread uniformly in the network, in line with [120] where the authors consider 5 test points for the producers. The content catalog we take into account is composed of 10^7 objects partitioned into 100 popularity classes, where the average object size is 1 Mbyte, as done in [39, 58]. Each consumer requests 10 objects in each time slot, in agreement with [122]. Object requests are distributed according to the rank-shift model discussed in Sec. 5.3, and we consider the ρ exponent in the 10^{-4} to 0.99 range, while the alpha exponent of the Zipf distribution can either be $\alpha = 0.8$ or $\alpha = 1.2$, as done in [77] and in line with [67, 155]. We consider different cache sizes, such that they can store from 1 to 5% of the total number of objects available, as considered in other studies (e.g., [39, 67]). The number of test points for the CDN nodes we deploy is up to 10, whereas we consider a reloca-



(a) *Netrail Cache Size,* $\alpha = 0.8, \rho = 0.01$

(b) *Netrail Cache Size,* $\alpha = 1.2, \rho = 0.01$

(c) *Netrail Cache Size,* $\alpha = 1.2, \rho = 10^{-4}$

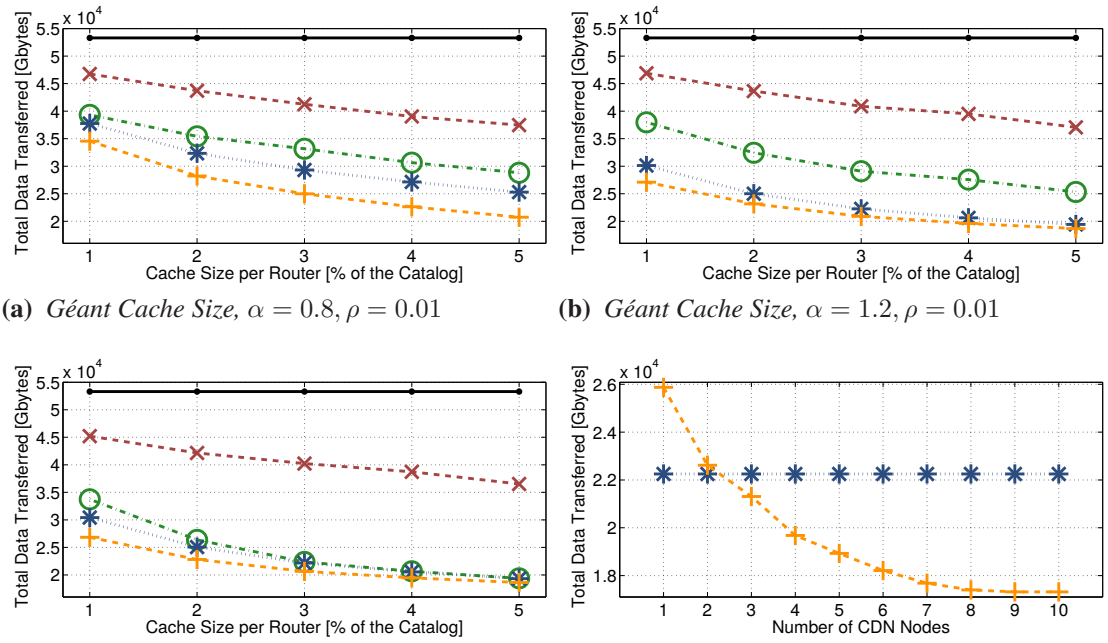
(d) *Legend*

Figure 5.5: Netrail Topology, Effect of the Cache Size. *Figures 5.5a-5.5c show the effect of the cache size in the Netrail topology, for different values of the Zipf* α *exponent, as well as the popularity evolution probability* ρ*. The legend of Fig. 5.5d is common to all the plots in Fig. 5.5-5.7.*

tion time between 1 and 15, meaning that in our analysis the CDN might be "as fast as" NDN or up to 15 times slower than that. If not stated otherwise, 3 CDN nodes will be deployed and they will have a relocation time equal to 3. We believe that these values can be used to perform a *fair* comparison of NDN and CDN: by deploying such a small number of CDN nodes we do not bias the analysis in favor of this latter architecture. The performance metric we consider is the total network traffic, since it is the objective function we take into account for the optimization models.

The behavior of the Netrail topology as a function of the cache size is shown in Fig. 5.5a (for $\alpha = 0.8, \rho = 0.01$), Fig. 5.5b (for $\alpha = 1.2, \rho = 0.01$) and Fig. 5.5c (for $\alpha = 1.2, \rho = 10^{-4}$). Considering the *No-Cache* curves for different combinations of α and ρ values (as in Fig. 5.5a-5.5c), we observe that a network that does not have caching functionalities leads to the same overall traffic (about 3.34 Tbytes for Netrail), even for different popularity evolution parameters. By introducing caching functionalities, the total network traffic can be reduced significantly: in the Netrail topology, caching reduces traffic up to 46% (Fig. 5.5c, CDN-Optimal caching policy, 5% cache size), while in the Géant topology, traffic savings can reach 66% (Fig. 5.6c, CDN-Optimal caching policy, 5% cache size).

An interesting observation on the random caching policy is that it leads to a total traffic that is rather independent with respect to the popularity evolution parameters α and ρ: the total traffic for NDN-Random is on average 8% lower than for No-Cache.



(a) *Géant Cache Size,* $\alpha = 0.8, \rho = 0.01$ **(b)** *Géant Cache Size,* $\alpha = 1.2, \rho = 0.01$

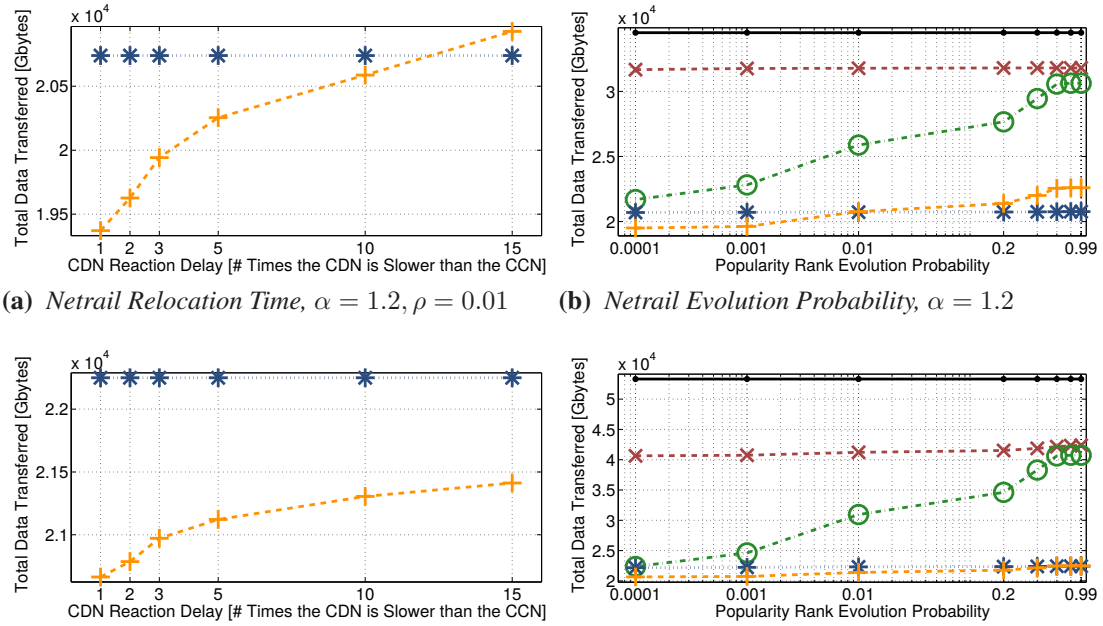
(c) *Géant Cache Size,* $\alpha = 1.2, \rho = 10^{-4}$ **(d)** *Géant CDN Nodes,* $\alpha = 1.2, \rho = 0.01$

Figure 5.6: Géant Topology, Effect of the Cache Size. *Figures 5.6a-5.6c show the effect of the cache size in the Géant topology, for different values of the Zipf* α *exponent, as well as the popularity evolution probability* ρ*. Figure 5.6d shows the effect of the number of CDN nodes in Géant.*

On the contrary, the LFU caching policy is very sensitive to the value of ρ: the lower the speed at which the popularity evolves, the better the objective function is, as shown in Fig. 5.5b and 5.5c. In particular, while in Fig. 5.5b LFU scores on average 25% traffic reduction compared to No-Cache, in Fig. 5.5c the same performance gain raises to 35%.

The topology size has a strong impact on the results, in particular, caching is more beneficial in larger topologies. In fact, if we compare the No-Cache curves of Fig. 5.5a and 5.6a, we observe 38% more traffic in Géant than Netrail. However, if we assume that caching functionalities are deployed in the network, as in the CDN-Optimal case, Géant requires on average only 5% more traffic than Netrail when $\alpha = 0.8$, and less than 8% more, when $\alpha = 1.2$ as in Fig. 5.5a,5.6a and 5.5b,5.6b.

Another remarkable result regards the efficiency of the different distribution architectures for different topologies. As a matter of fact, while in the small Netrail topology NDN and CDN almost exhibit the same performance, in Géant there is a large gap between the two solutions: on average, CDN reduces the total network traffic 14% more than what NDN does, leading to an overall 51% performance gain (on average), with respect to the total traffic transmitted in the No-Cache scenario, when $\alpha = 0.8$. This behavior is caused by the fact that the optimal solution of the NDN network tends to scatter many copies of the same popular objects in all the caching nodes, whereas cache duplication is significantly reduced by aggregating the storage on few CDN surrogates



(a) *Netrail Relocation Time,* $\alpha = 1.2, \rho = 0.01$ **(b)** *Netrail Evolution Probability,* $\alpha = 1.2$

(c) *Géant Relocation Time,* $\alpha = 1.2, \rho = 0.01$ **(d)** *Géant Evolution Probability,* $\alpha = 1.2$

Figure 5.7: Effect of the Relocation Time, Effect of the Evolution Probability. *Figure 5.7a and 5.7c show the effect of the relocation time in Netrail and Géant, respectively. The effect of the evolution probability on the overall traffic is instead shown in Figures 5.7b and 5.7d in Netrail and Géant, respectively.*

occupying central locations in the network topology (*k-median* positioning). This key finding is even more evident in Fig. 5.6d, where we observe that few CDN nodes in the Géant topology (only 3 in the considered scenario) are sufficient to make CDN reach better performance values than those obtained with NDN.

The sensitivity of the objective function to the relocation time $|\mathcal{T}_r|$ is depicted in Fig. 5.7a and 5.7c, for the Netrail and Géant topology, respectively. In both cases we deploy 3 CDN nodes. In the Netrail topology (Fig. 5.7a), even if we assume that CDN nodes are 10 times slower to react to popularity changes than the NDN counterpart, CDN still shows better performance. On the other hand, for the Géant topology (Fig. 5.7c), CDN always leads to better performance even setting $|\mathcal{T}_r| = 15$.

Also the ρ parameter that drives the speed of the content popularity evolution affects the objective function, as depicted in Fig. 5.7b for Netrail and Fig. 5.7d for Géant. The LFU policy is the one that is subject to the largest variation: by increasing the ρ value we reduce the locality of reference of the requests, and this, in turn, penalizes the LFU strategy. It is very interesting to note that in the Netrail topology, when $\rho < 0.01$ CDN is to be preferred, whereas for larger ρ values, NDN leads to the best performance value.

5.6 Related Work

Adhikari et al. study in [12] the Netflix video streaming platform, by analyzing the video delivery performance in a scenario where many CDNs are used for video streaming purposes. In [127], Mansy and Ammar focus on a scenario where the CDN cooperates with P2P to serve *adaptive* video streaming requests. The authors show that such a hybrid scenario has interesting performance properties that can potentially reduce the costs paid by the content provider. Liu et al. have measured in [122] the performance of video distribution when clients leverage the CDN infrastructure to retrieve video content. In order to further improve the users' *quality of experience*, the authors suggest to adopt a control plane that automatically selects the best bit-rate and CDN server according to the network state.

In terms of performance optimization, three classic problems have to be solved in a CDN:

1. *Server Placement*: choose the locations where to place the CDN servers.
2. *Replica Object Placement*: choose the locations and the number of object replicas by distributing them on the available servers.
3. *Surrogate Server Selection*: route object requests by selecting a replica server storing a copy of the given object.

The *center placement problem* is used to solve both server placement as well as the replica object placement, by modeling the problem as a graph where a given distance metric should be minimized [102, 148]. The size of the problem, which becomes even larger when studying object placement, forces to formulate heuristic algorithms that find sub-optimal solutions, such as those presented in [55, 113].

Finally, in [173], surrogate server selection strategies used by YouTube are evaluated in order to understand which parameters may influence the CDN server selection process.

Compared to previous literature, in this work we compared the performance gains that NDN and CDN can achieve by means of using optimization models and by considering a time-varying content popularity. Our key findings are line with relevant research works, such as [67, 156].

Fayazbakhsh et al. study in [67] the performance of NDN specifically comparing the improvements achievable with an *edge-based caching* with respect to a *full-fledged* NDN. In particular, in line with our key findings, they show that most of the performance benefits can be gained by deploying caching storage only on the edge nodes, whereas by distributing the memory also on the other nodes they can achieve a limited 4% performance gain.

Thorough simulation campaigns are presented in [156], where Rossini and Rossi study the performance of NDN focusing on forwarding strategies and caching policies. They show that limited benefits can be achieved by adopting complex caching policies: randomized caching decisions can perform as well as more complex ones, while significantly reducing the computational overhead that they introduce. Moreover, they also show that multi-path forwarding capabilities may play against the network efficiency.

5.7 Conclusion

In this chapter we compared NDN and CDN by formulating novel optimization models to analyze the performance bounds of these network architectures, considering time-varying demands to represent content popularity evolution. The proposed models are such that the relevant characteristics of NDN and CDN are explicitly taken into account.

Our numerical results suggest that the performance bounds for CDN architectures are better in terms of network traffic than observed for NDN, even when few CDN replica servers are deployed in the network. In our considered scenarios, we observed that in the Netrail topology NDN can reach a better value for the objective function compared to CDN only if this latter is at least 15 times slower to react to popularity changes. On the other hand, while considering the larger Géant topology, we observed that CDN is always to be preferred since it reaches up to 14% better performance than NDN.

Our analysis is in line with the results presented in [67]: spreading the caching storage uniformly in the network does not necessarily lead to better performance than that obtained by concentrating it on fewer nodes. In other words, in terms of the overall network traffic NDN does not necessarily perform better than CDN. Our view is that rather than being two antagonist models, NDN and CDN should complement each other; in particular NDN is a very promising architecture to reduce the management overhead that the network introduces, whereas CDN is to be preferred if the main goal is to achieve the highest performance gains.

CHAPTER 6

Stochastic Planning of Virtual Content Delivery Networks

As discussed in Sec. 4.1, the first prototypes for routers supporting Named-Data Networking have begun to appear. However, despite the fact that NDN can boost the content distribution capabilities of the network, its worldwide adoption demands to change the protocol stack used by both the end-points as well as intermediate network nodes and, for this reason, it will unlikely happen in the near future. Notwithstanding, content distribution still deserves novel techniques to efficiently support network content delivery in a cost-effective manner. In particular, in this chapter, rather than choosing a clean-slate approach, we advocate for an evolutionary proposal for CDNs, that is compatible with the current TCP/IP protocol stack and does not require any change in the end-points.

Content Delivery Networks (CDNs) have been identified as one of the relevant use cases where the emerging paradigm of Network Functions Virtualization (NFV) will likely be beneficial. In fact, virtualization fosters flexibility, since on-demand resource allocation of virtual CDN nodes can accommodate sudden traffic demand changes. However, there are cases where physical appliances should still be preferred, therefore we envision a mixed architecture in between these two solutions, capable to exploit the advantages of both of them.

Motivated by these reasons, in this chapter we formulate a two-stage stochastic planning model that can be used by CDN operators to compute the optimal long-term network planning decision, deploying physical CDN appliances in the network and/or leasing resources for virtual CDN nodes in data centers. Key findings demonstrate that for a large range of pricing options and traffic profiles, NFV can significantly save

network costs spent by the operator to provide the content distribution service.

6.1 Introduction

The worldwide success of content-rich web applications like social networks or on-demand streaming services has forced network operators to invest a significant amount of money in order to keep up-to-date their communication infrastructures [12]. In particular, Content Delivery Networks (CDNs) have nowadays become a necessary (and well-established) technology to efficiently serve the traffic demands that consumers are generating, while supporting the high level of performance and reliability that providers are demanding [121].

Although CDN is an effective infrastructure to move replicas of popular contents closer to the users' locations, it requires significant investments to be built and operated. As an example, the Akamai infrastructure comprises more than 61 000 servers deployed in 1 000 networks and 70 countries worldwide [140]. To reduce the capital expenditures and improve the performance of CDNs, organizations such as the Internet Engineering Task Force (IETF) and the European Telecommunications Standards Institute (ETSI) have recently begun a standardization process for two alternative architectures:

— Content Delivery Network Interconnection (CDNI);

— Virtual Content Delivery Network (vCDN).

Despite the fact that they both have to deal with network content distribution, these proposals have a radically different scope: the former (CDNI) is mostly concerned with the co-operation of many CDN providers [138], whereas the latter (vCDN) proposes to virtualize the CDN services on top of the novel layer for Network Functions Virtualization (NFV) [65].

While both the architectures aim at optimally exploiting available physical resources, the grounds of CDNI are settled on agreements between different CDN operators that often are in direct competition in the same market. On the other hand, the NFV approach is to run network functions in a virtualized environment, executed on a shared physical infrastructure composed of industry standard high volume servers, storage and switches [65]. Therefore, vCDN implemented on top of NFV enjoys the positive advantage of avoiding potential competition issues, since the virtualized environment ensures the necessary level of isolation between the different network functions. Furthermore, spare NFV substrate capacity can be leased by network operators to third parties, a condition that makes vCDN appear even more profitable.

Motivated by the previous background, in this chapter we tackle a fundamental issue that arises in such context: the planning problem for a mixed physical-virtual Content Delivery Network under uncertain traffic demands. In our formulation, the CDN operator can choose between *purchasing* physical CDN appliances and *leasing* instances of virtual CDN nodes provided by an infrastructure operator. However, while vCDN nodes can be activated on-demand if the traffic requests require to do so, the installation of physical CDN nodes must be chosen on a long-term schedule. For both physical or virtual CDN surrogate servers, the operator must carefully choose their location, while minimizing the overall costs. Due to the fact that planning is performed on a long-term

basis, the theoretical framework of *stochastic optimization* will be used to guarantee robustness of the solution with respect to the uncertainty in the probabilistic description of future traffic demands.

The contribution of this chapter is summarized as follows:

1. We formulate a two-stage stochastic planning model used by CDN operators to compute the optimal, long-term network planning decision, under traffic demands uncertainty.
2. We propose a greedy heuristic approach that finds good solutions (close to the optimum, in several cases) even for large-scale network topologies, and we compare its execution time with *exact* solution strategies: (1) the deterministic equivalent program in the extensive form and (2) the L-shaped algorithm (*single* and *multicut* versions) [177].
3. We perform an extensive numerical evaluation, considering real scale topologies and a wide range of parameters.

Our key findings suggest that a mixed physical-virtual CDN infrastructure leads to significant lower costs when compared to those obtained by a standard CDN, while being robust with respect to sudden traffic demand changes.

The chapter is organized as follows: Sec. 6.2 presents our contribution, including the system model we consider (Sec. 6.2.1), the optimization model (Sec. 6.2.2), and the greedy algorithm (Sec. 6.2.3). Numerical results are presented in Sec. 6.3. Related works are discussed in Sec. 6.4, whereas Sec. 6.5 concludes the chapter.

6.2 Optimal Content Delivery in NFV

In this section we describe our proposed solution for the optimal content delivery planning in NFV. Sec. 6.2.1 introduces the system model and relevant assumptions. In Sec. 6.2.2 we formulate the optimization model, while in Sec. 6.2.3 we discuss the design of a near-optimal heuristic algorithm.

6.2.1 System Model and Assumptions

Figure 6.1 shows the system model we consider in our proposal. In this work we tackle the long-term planning problem from the point of view of a CDN provider. The aim of the provider is to perform two choices:

1. Select *whether* and *where* physical CDN nodes should be installed in the network topology;
2. Select the optimal *request routing*, given the installed physical CDNs and the virtual nodes available.

Since the planning decision is operated on a long-term time schedule, the provider does not deterministically know what is going to happen in the future. On the other hand, we assume that an estimate of the continuous probability distribution of future traffic demands is known for the planning problem. However, for the sake of simplicity, and as frequently done in the literature (e,g: [63, 177]), we discretize this information on

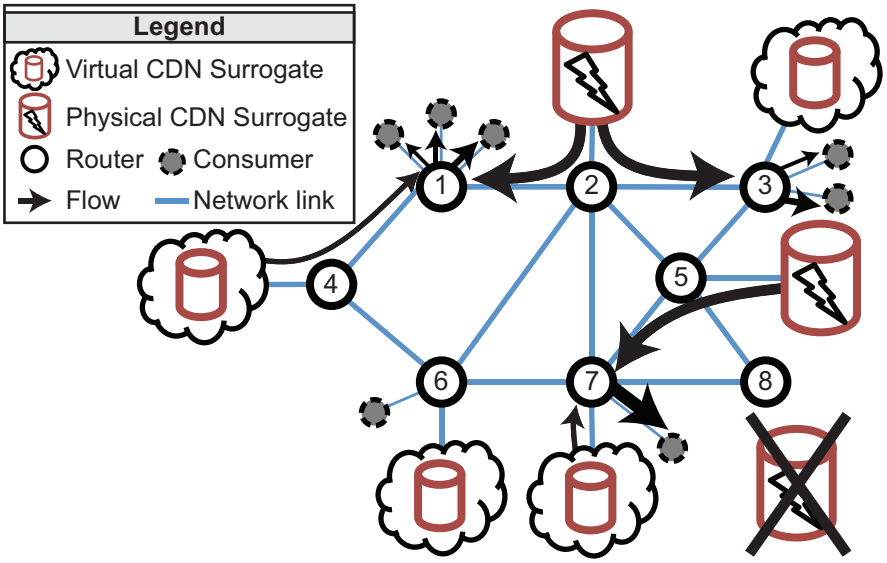


Figure 6.1: System model. *The network is composed by consumers, routers, virtual and physical CDN surrogates. Our proposed optimization model selects (1) the planning of physical CDNs and (2) request routing.*

a finite number of scenarios, therefore our traffic model is jointly *time-varying* and *stochastic* in its nature.

As shown in Fig. 6.1, traffic demands are expressed by the consumers. Virtual and physical CDN nodes can both be used to efficiently serve consumers' demands, however there exist major differences (in terms of capacity, activation choice and pricing policy) between these two types of CDN hosts:

— *Capacity:* virtual CDN nodes can serve a lower amount of traffic requests since the presence of the hypervisor and the shared hardware infrastructure reduces the throughput of the CDN surrogates.

— *Activation choice:* virtual CDN servers can be used on-demand, and they do not need to be explicitly activated. On the other hand, if the operator chooses to install a physical CDN server, it will be activated once and it will stay active throughout the entire time horizon.

— *Pricing policy:* physical CDN nodes have an activation price that considers both the capital expenditure (CAPEX) for the acquisition of the device as well as the long-term operational expenditure (OPEX) costs. On the other hand, the virtual CDN nodes have a traffic-proportional price related to the OPEX cost component, which is the per-bandwidth leasing price that the vCDN owner charges.

Since virtual CDN nodes can be used on-demand, they can serve the portion of traffic requests with the highest variability. To improve the quality of service of a CDN, surrogate servers must be selected close to the location of consumers. For this purpose, we use the link delay to control the performance of the infrastructure: we assume that the content provider wants to serve a fraction of the overall requests within a bounded limit on the delay.

Table 6.1: *Notation used in this chapter.*

Input Parameters	
$\mathcal{D}$	Set of consumers (*destination* nodes)
$\mathcal{S}$	Set of candidate surrogate servers (*source* nodes) $\mathcal{S} = \mathcal{S}^P \cup \mathcal{S}^V$
$\mathcal{S}^P$	Set of candidate physical CDN servers
$\mathcal{S}^V$	Set of candidate virtual CDN servers
$\mathcal{T}$	Set of time slots
Φ	Set of stochastic scenarios
$r_d^{t,\phi}$	Traffic requests of client $d \in \mathcal{D}$, at time slot $t \in \mathcal{T}$, for scenario $\phi \in \Phi$
ϵ	Minimum service level guaranteed (fraction of traffic requests served with a bounded delay of at most Δ)
Δ	Maximum tolerated delay
$\delta_{s,d}$	Delay between the nodes $s \in \mathcal{S}$, $d \in \mathcal{D}$
$\mathcal{K}_s^P$	Capacity of the physical CDN server $s \in \mathcal{S}^P$
$\mathcal{K}_s^V$	Capacity of the virtual CDN server $s \in \mathcal{S}^V$
$\mathcal{C}_s^P$	CAPEX and OPEX costs of the physical CDN server installed at $s \in \mathcal{S}^P$
$\mathcal{C}_s^V$	Usage cost of the virtual CDN server at the candidate site $s \in \mathcal{S}^V$
p_ϕ	Realization probability for the scenario $\phi \in \Phi$

Decision Variables	
a_s	0-1 Physical CDN activation, $a_s = 1$ if a physical CDN is installed at $s \in \mathcal{S}^P$
$y_{s,d}^{t,\phi}$	Physical CDN flow variable for requests served by $s \in \mathcal{S}^P$ to client $d \in \mathcal{D}$, at time $t \in \mathcal{T}$, and scenario $\phi \in \Phi$
$z_{s,d}^{t,\phi}$	Virtual CDN flow variable for requests served by $s \in \mathcal{S}^V$ to client $d \in \mathcal{D}$, at time $t \in \mathcal{T}$, and scenario $\phi \in \Phi$

Popular CDN providers such as Amazon CloudFront or Microsoft Azure CDN do not have an activation cost but charge for their services according to the amount of traffic that surrogates are providing, regardless of the caching storage used. Moreover, frequent flash crowds make the object popularity suddenly change, whereas having an estimate of the aggregate future demands is instead much easier [68]. For these reasons, we focus on infrastructure planning and request routing, while we do not tackle the replica placement problem.

6.2.2 Optimization Model

In this section we describe the optimization model we formulate for the optimal planning of a mixed physical-virtual CDN infrastructure. The notation is summarized in Table 6.1.

Let $\mathcal{S} = \mathcal{S}^P \cup \mathcal{S}^V$ be the set of CDN surrogate servers, where $\mathcal{S}^P$ and $\mathcal{S}^V$ represent candidate physical and virtual surrogate nodes, respectively. Consumers are denoted with $\mathcal{D}$, the set of time slots is represented with $\mathcal{T}$, while the set of stochastic scenarios is Φ. Each scenario $\phi \in \Phi$ has an associated realization probability, represented by

p_ϕ. Consumers $d \in \mathcal{D}$ express a time-varying traffic demand for each scenario $\phi \in \Phi$, that we indicate with $r_d^{t,\phi}$. The CDN provider ensures that at least a fraction ϵ of the aggregate requests in every time slot is served by CDN nodes within a bounded delay, denoted as Δ. The topological information is encoded in our proposed optimization model using the $\delta_{s,d}$ input parameter, which represents the delay between client $d \in \mathcal{D}$ and CDN node $s \in \mathcal{S}$. $\mathcal{K}_s^P$ and $\mathcal{K}_s^V$ are the bandwidth capacities for physical and virtual CDNs, respectively. Physical nodes have an activation cost $\mathcal{C}_s^P$, while virtual CDN nodes have a traffic-proportional cost $\mathcal{C}_s^V$.

Our proposed optimization model chooses the optimal physical nodes placement and request routing. a_s is a binary decision variable that is set to 1 if and only if the physical candidate server $s \in \mathcal{S}^P$ is activated. Traffic requests for consumer d, in time slot t for scenario ϕ can be served by flows $y_{s,d}^{t,\phi}$ and $z_{s,d}^{t,\phi}$. In particular, $y_{s,d}^{t,\phi}$ is a flow originating from the physical node $s \in \mathcal{S}^P$, while $z_{s,d}^{t,\phi}$ is a flow provided by the virtual node $s \in \mathcal{S}^V$.

The deterministic equivalent program in the extensive form for the CDN planning problem of our infrastructure (EF-CDN) is formulated as follows:

$$\min \sum_{s \in \mathcal{S}^P} \left[\mathcal{C}_s^P a_s \right] + \mathbb{E}_\Phi \left[\sum_{s \in \mathcal{S}^V} \sum_{t \in \mathcal{T}} \sum_{d \in \mathcal{D}} \left(\mathcal{C}_s^V z_{s,d}^{t,\phi} \right) \right] \tag{6.1}$$

subject to:

$$\sum_{d \in \mathcal{D}} y_{s,d}^{t,\phi} \leq a_s \mathcal{K}_s^P \qquad \forall s \in \mathcal{S}^P, t \in \mathcal{T}, \phi \in \Phi \tag{6.2}$$

$$\sum_{d \in \mathcal{D}} z_{s,d}^{t,\phi} \leq \mathcal{K}_s^V \qquad \forall s \in \mathcal{S}^V, t \in \mathcal{T}, \phi \in \Phi \tag{6.3}$$

$$\sum_{s \in \mathcal{S}^P} y_{s,d}^{t,\phi} + \sum_{s \in \mathcal{S}^V} z_{s,d}^{t,\phi} = r_d^{t,\phi} \qquad \forall d \in \mathcal{D}, t \in \mathcal{T}, \phi \in \Phi \tag{6.4}$$

$$\frac{\sum_{d \in \mathcal{D}} \left(\sum_{s \in \mathcal{S}^P | \delta_{s,d} \leq \Delta} y_{s,d}^{t,\phi} + \sum_{s \in \mathcal{S}^V | \delta_{s,d} \leq \Delta} z_{s,d}^{t,\phi} \right)}{\sum_{d \in \mathcal{D}} r_d^{t,\phi}} \geq \epsilon \qquad \forall t \in \mathcal{T}, \phi \in \Phi \tag{6.5}$$

$$a_s \in \{0, 1\} \qquad \forall s \in \mathcal{S}^P \tag{6.6}$$

$$y_{s,d}^{t,\phi}, z_{s,d}^{t,\phi} \in \mathbb{R}^+ \qquad \forall s \in \mathcal{S}, d \in \mathcal{D}, t \in \mathcal{T}, \phi \in \Phi. \tag{6.7}$$

The objective function (6.1) minimizes the overall costs given by the activation of physical CDN nodes as well as the usage of the virtual infrastructure. In particular, the virtual cost component is computed as the expected value for all the considered scenarios. Constraints (6.2) set a capacity bound on the overall demand served by physical CDN surrogates. If a physical surrogate is not activated, that is $a_s = 0$, it will

Algorithm 10: Heuristic solver

Input : $\langle \mathcal{D}, \mathcal{S}, \mathcal{T}, \Phi, r_d^{t,\phi}, \mathcal{K}_s^P, \mathcal{K}_s^V, \mathcal{C}_s^P, \mathcal{C}_s^V, \delta_{s,d}, \Delta, \epsilon, p_\phi \rangle$

Output: $\hat{a}, \hat{y}_{s,d}^{t,\phi}, \hat{z}_{s,d}^{t,\phi}$, min_cost

$\hat{a} = [1; 1; ...; 1]$; phy_nodes = *sort_phy_CDN_nodes()*;

if $\neg$*is_feasible*($\hat{a}$) **then**

  return INFEASIBLE_ASSIGNMENT ;

end

$\langle \hat{y}_{s,d}^{t,\phi}, \hat{z}_{s,d}^{t,\phi}$, min_cost$\rangle$ = *get_LP_solution*($\hat{a}$); best_sol = $\hat{a}$;

foreach $s \in$ phy_nodes **do**

  $\hat{a}_s = 0$;

  $\langle \hat{y}_{s,d}^{t,\phi}, \hat{z}_{s,d}^{t,\phi}$, current_min_cost$\rangle$ = *get_LP_solution*($\hat{a}$);

  if current_min_cost $\geq$ min_cost **then**

    break ;

  end

  min_cost = current_min_cost ; best_sol = $\hat{a}$;

end

$\hat{a}$ = best_sol ; $\langle \hat{y}_{s,d}^{t,\phi}, \hat{z}_{s,d}^{t,\phi}$, min_cost$\rangle$ = *get_LP_solution*($\hat{a}$);

not be capable to serve any request. Similarly, the virtual CDN nodes capacity is fixed in (6.3). In (6.4) we make sure that the overall clients' demands are served in any time slot and scenario, by virtual or physical surrogate servers. Flows can be split across multiple CDN servers. In (6.5) we control the overall service quality. We make sure that a fraction of at least ϵ requests in each time slot is served by CDN surrogates within a maximum delay of Δ. Finally, binary restrictions on the activation variables are set in (6.6), while non-negativity constraints on the continuous flow variables are enforced in (6.7). Rather than considering the worst or mean case, the stochastic formulation ensures that constraints hold in every scenario, while the objective function is optimized given the uncertainty on future traffic requests.

In order to solve the optimization problem (6.1)-(6.7) we employ different strategies:

1. A mixed integer linear programming solver (MILP);
2. The L-shaped algorithm (*single* and *multicut* versions);
3. A polynomial-time greedy heuristic.

In Sec. 6.2.3 we present our proposed heuristic, while we refer to Van Slyke and Wets [177] for an introduction to the L-shaped algorithm since, for the sake of brevity, in this chapter we omit its implementation details.

6.2.3 Heuristic Solver

The L-shaped algorithm can be used to find an exact solution of the optimization problem we formulated in Sec. 6.2.2. Theoretical results guarantee that the L-shaped algorithm converges to the optimal solution, but in some cases the speed of convergence might be too slow for the considered application. In this section we describe the

polynomial-time heuristic we designed to compute a close-to-optimal solution to the planning problem. Pseudo-code is provided in Algorithm 10 to describe the steps we use to achieve this purpose.

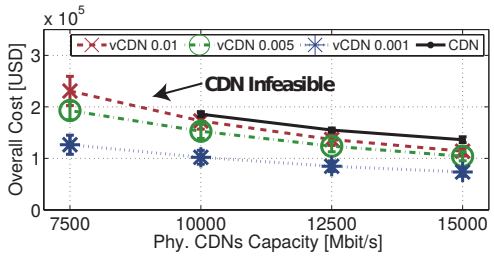
At the beginning of the algorithm, in Step 1, all the physical nodes are activated and sorted. In particular, we give higher priority to those nodes that can serve the largest amount of traffic demands within a delay of Δ. Given the structure of the problem, infeasible solutions are those that cannot be served even when all the physical CDN nodes have been activated. In Step 2 we check this condition and we eventually signal a potential infeasibility. The *get_LP_solution* function computes a solution for the continuous relaxation of the model (6.1)-(6.7), using a standard linear programming solver, and therefore has a polynomial-time complexity. As outputs, it returns the optimal flows for the physical and virtual CDN nodes as well as the overall cost. In case of infeasibility, the output value of *min_cost* is set to infinite. The loop in Step 3 deactivates at every iteration one new physical node, according to the previously generated ordering, and it completes when the objective function does not improve anymore. Lastly, in Step 4 we compute the optimal flows starting from the best physical nodes allocation choice.

6.3 Numerical Results

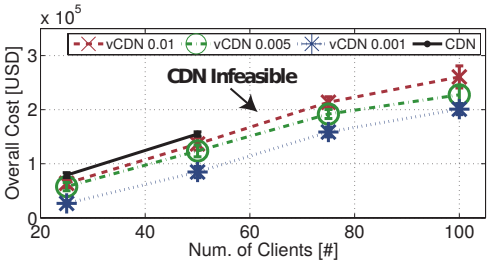
In this section we present the numerical results we obtained performing a thorough analysis of our models and heuristics under realistic network conditions.

Unless stated otherwise, our network topology is created using the Barabási–Albert model and is composed of 50 consumers, 20 physical and 15 virtual CDN nodes. Traffic demands are generated using as a reference the Cisco VNI data for the 2014-2017 years: the planning horizon is of 3 years and we used 36 different time slots. Traffic uncertainty is taken into account by considering 10 different scenarios, with a variable overall demand uniformly distributed between 80% and 120% of the Cisco forecast. Slightly better results were observed considering the more favorable case of the binomial distribution with success probability $p = 0.5$. To control the overall demand, we limit to 20 Gbit/s the maximum traffic requests that a consumer can generate in a time slot of a scenario. Physical CDN nodes have a capacity of 12.5 Gbit/s, while virtual CDNs can serve up to 8 Gbit/s. Similar trends have been observed for other values of the capacity, omitted here for the sake of brevity. Link delays are generated in the range of those available on Rocketfuel for the Sprintlink (US) topology, with an average delay of 3 ms. Moreover, we assume the CDN provider wants to guarantee that at least 95% of the requests are served by surrogates with a delay lower than 12 ms, that is, selected CDN nodes are, on average, 4 hops far from the consumer's location.

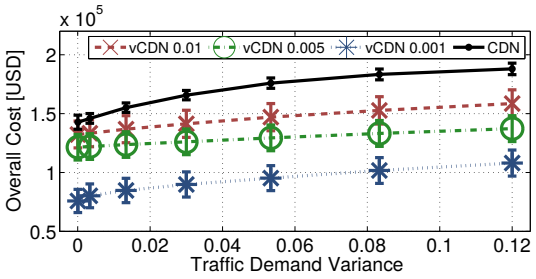
Lastly, prices are set as follows: we assume that the cost to install and operate one CDN node is set in the range $[8; 12]$ kUSD, while different prices will be considered for virtual CDN nodes in the range $[0.001; 10]$ USD per Mbit/s. For the same physical-virtual price ratios, even by considering different values for the physical CDN pricing, we observed similar trends as those discussed in this section. Hereafter, we report the result we obtained using CPLEX 12.5 as a MILP solver, bounding the maximum execution time of the algorithms to 1 hour (with a 5% MIP gap), and using a machine



(a) *Effect of physical CDN capacity*



(b) *Effect of the number of clients*



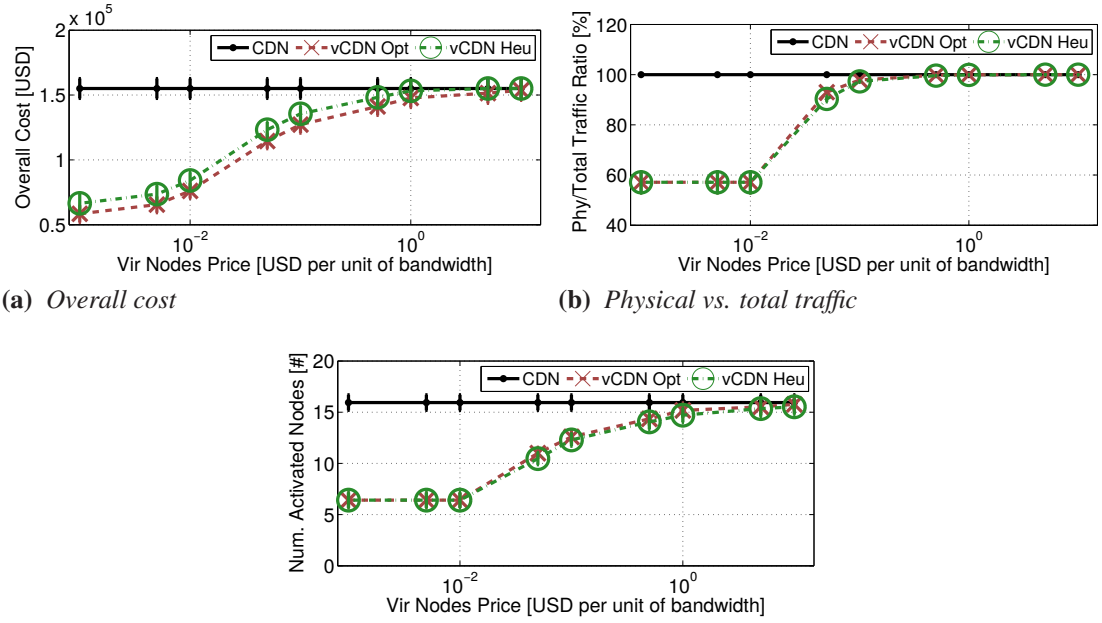
(c) *Effect of traffic variance*

Figure 6.2: NFV Benefits. *Plots 6.2a-6.2c show the cost benefits of an architecture composed of a mix of physical and virtual CDN nodes (denoted in the figures with vCDN), with respect to the scenario where only the physical CDN infrastructure is used (denoted in the figures with CDN). Different prices are considered for the vCDN case, as shown in the legend.*

equipped with a quad-core Intel i7-3770 (3.40 GHz) CPU with 16 Gbyte of RAM. Lastly, for each of the results we performed 20 different runs and we report the narrow 95% confidence intervals.

NFV Benefits. Fig. 6.2 shows the cost benefits for the mixed physical-virtual CDN architecture, considering different virtual prices. As expected, in Fig. 6.2a-6.2c, lower CDN prices lead to lower overall costs. The effect of the physical CDN capacity is shown in Fig. 6.2a. If the capacity of the physical CDN appliances is lower than 10 Gbit/s, using virtual nodes becomes mandatory since otherwise an infeasibility is produced. When the physical CDN nodes capacity is set to 15 Gbit/s, cost savings up to 46% are experienced for cheap virtual CDN pricing (i.e, 0.001 USD per unit of bandwidth), whereas the saving is reduced to 16% if we set a vCDN price of one order of magnitude larger (i.e, 0.01 USD per unit of bandwidth). Fig. 6.2b shows the effect of the number of clients on the costs. The physical-only CDN infrastructure cannot handle more than 50 clients, whereas up to 100 clients can be served if we also leverage the 15 virtual CDNs deployed. The effects of traffic uncertainties are quantified in Fig. 6.3c, where we show the overall cost as a function of the traffic demand variance. With the largest variance that we took into account, the vCDN infrastructure leads to cost savings in the range 16-43% according to the vCDN pricing.

Effect of the vCDN Price. Due to its remarkable effect, in Fig. 6.3 we show the impact of the virtual CDN pricing using the different solution algorithms. The overall cost is portrayed in Fig. 6.3a. Solutions obtained with exact solvers such as the



(a) *Overall cost*

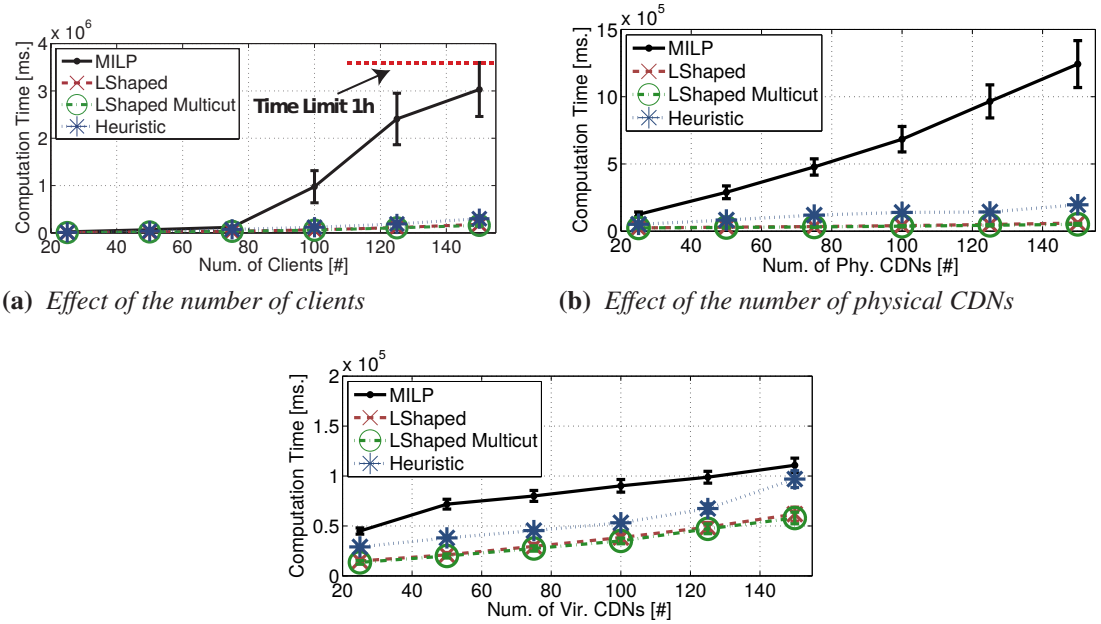
(b) *Physical vs. total traffic*

(c) *Physical nodes activation*

Figure 6.3: Effect of the vCDN Price. *Plots 6.3a-6.3c show the effect of the virtual nodes pricing on the overall cost (Fig. 6.3a), the fraction of traffic served by the physical infrastructure (Fig. 6.3b) and the number of physical nodes activated (Fig. 6.3c), for different solution algorithms as well as a physical-only infrastructure (denoted in the figures with CDN).*

deterministic equivalent program in the extensive form or the L-shaped algorithm lead to costs up to 11% (and on average 6%) lower than those reported with the heuristic. Fig. 6.3a clearly shows that the higher the prices, the lower the economic benefits of using a mixed physical-virtual CDN infrastructure. In particular, considering prices in the range $[0.001; 0.5]$ USD per unit of bandwidth, the cost savings compared to the physical-only solution are in the range 5-64%. Fig 6.3b shows the proportion of physical traffic with respect to the overall demand, as a function of the price of virtual nodes. Cheap prices make the virtual CDN capacity be fully saturated, and for this reason the left hand-side of Fig. 6.3b has an horizontal trend that accounts for 57% of the overall traffic. As a consequence, in Fig. 6.3c is reported that the number of physical CDN nodes does not increase for virtual prices lower than 0.01 USD. Comparing Fig. 6.3b and 6.3c, the slope of the curve is less steep in the second plot, since there are cases where it is convenient to strategically deploy a physical CDN server in a special position of the topology even though it is not fully used by the clients. It is interesting to note that for both Fig. 6.3b and 6.3c the heuristic algorithm leads to solutions that are practically overlapped to the optimal choice.

Computing time. To limit the effects of infeasibilities that negatively affects results on the computing time, hereafter we raise the number of virtual CDN nodes to 50, making the network be capable to serve up to 150 consumers. Fig. 6.4 shows the execution time of the different algorithms as a function of the number of clients (Fig. 6.4a), the number of physical (Fig. 6.4b), and virtual CDNs (Fig. 6.4c).



(a) *Effect of the number of clients*

(b) *Effect of the number of physical CDNs*

(c) *Effect of the number of virtual CDNs*

Figure 6.4: Execution time. *Plots 6.4a-6.4c show the behavior of the different solution algorithms as a function of the number of clients as well as the number of surrogate nodes. We observe that the number of clients has the most remarkable effect on the execution time.*

The number of clients is the parameter that mostly affects the execution time, as portrayed in Fig. 6.4a. In particular, the MILP solver for the deterministic equivalent program has a time trend that is exponential in the number of served clients (Fig. 6.4a), but linear in the number of physical (Fig. 6.4b) and virtual CDNs (Fig. 6.4c). While the MILP solver can hardly scale to topologies with a larger number of nodes, this possibility is instead offered by the L-shaped decomposition and our proposed heuristic. As a matter of fact, all these algorithms can solve the planning problem saving up to 94% of time compared to MILP, when considering 150 clients, as shown in Fig. 6.4a. Lastly, although there are cases where the heuristic algorithm is slightly slower than the L-shaped algorithm (as in Fig. 6.4c), we remark the fact that the heuristic has a worst-case polynomial time complexity, whereas a comparable theoretical result for the L-shaped algorithm does not hold.

6.4 Related Work

In this section, we survey relevant literature on Network Functions Virtualization (Sec. 6.4.1), Content Delivery Networks (Sec. 6.4.2) and Stochastic Optimization techniques (Sec. 6.4.3).

6.4.1 Network Functions Virtualization

To support the network virtualization paradigm, one of the challenges that must be solved is to find a mapping between a set of requests for virtual network resources

and the available underlying physical infrastructure, ensuring that desired performance requirements on nodes and links are guaranteed [51]. This is the *virtual network embedding* (VNE) problem, which is known to be NP-hard, since it can be reduced to the multi-way separator problem [51]. VNE has received a lot of attention from the community, and several heuristic algorithms have been proposed, e.g. in [48, 52, 103].

Jarray et al. propose in [103] a column-generation technique coupled with a rounding heuristic to discover the most profitable embedding, under the constrained physical capacity of the infrastructure. Deterministic and randomized rounding techniques are used by Chowdhury et al. in [52], where they further facilitate the virtual link mapping by designing an augmented graph description to efficiently support node location constraints. Cheng et al. in [48] solve the node mapping step with a greedy algorithm: higher ranking is given to the nodes that possess more spare resources and are placed in better locations of the network.

Rather than assuming that the operator knows *a-priori* the traffic demands, our contribution is to consider the case where their probability distribution is known, and our proposed formulation is robust with respect to such uncertainty.

Botero et al. in [28] tackle the problem of virtual resource consolidation in the name of energy efficiency. By formulating an exact mixed integer linear programming (MILP) model, the authors observe that it is possible to save up to 30% of nodes energy consumption.

6.4.2 Content Delivery Networks

In the last few years, *content multihoming* is emerging as a novel technique for content delivery networks that makes it possible to jointly use many CDN services: [12, 121, 180].

Adhikari et al. show in [12] that the Netflix infrastructure already leverages multiple CDNs (Akamai, LimeLight and Level-3). The authors observe that the customers are mapped to a particular CDN in a rather static manner. In [121], Liu et al. further confirm that other major content publishers such as Hulu, Microsoft, Apple, Facebook and MSNBC are currently already exploiting content multihoming. Furthermore, given the practical relevance of this architecture, they design optimization algorithms to minimize the overall distribution costs under constrained quality requirements. Finally, Wang et al. extended in [180] the work of Liu et al., by explicitly considering capacity constraints on the surrogate nodes.

The novelty of our approach is to consider a new distribution architecture, implemented on top of NFV, where virtual CDN nodes can be used for content delivery purposes. Rather than forcing the provider to settle agreements with other CDN operators, our proposal guarantees better isolation and dramatically limits potential competition issues.

6.4.3 Stochastic Optimization

In practical scenarios, network design cannot assume that future traffic demands are known a-priori; on the other hand, more advanced optimization techniques must be

used to take into consideration the stochastic nature of input parameters: [20, 63, 123].

Atamtürk et al. formulate in [20] a two-stage network design model with traffic demand uncertainty. In their approach, the operator performs the planning decision according to a probabilistic description of traffic demands. The value of the second-stage recourse variables is chosen by changing flow routing. Liu describes in [123] the basic stochastic procedures applied to a flow assignment network design problem, showing the *here-and-know* solution and the *scenario-tracking* result obtained for the flow-assignment. A multistage stochastic programming model for mobile radio access networks has been proposed by Eisenblätter et al. in [63]. In their formulation they jointly take into consideration the coverage and capacity of their communication infrastructure.

In line with previous literature, our formulation takes into account the uncertainty embedded in future traffic demands. Our contribution is to apply the theoretical framework of stochastic optimization to content distribution.

We refer to Birge and Louveaux [24] for an introduction to stochastic programming modeling and solution techniques.

6.5 Conclusion

In this chapter we tackled the stochastic planning problem for content delivery to study potential benefits that Network Functions Virtualization can lead for content distribution purposes. We considered a mixed architecture where both physical as well as virtual CDN nodes can be used by a CDN owner to implement the content distribution service. The owner performs the planning choice for the physical CDN infrastructure on a long-term time schedule, possessing only a stochastic estimate of future traffic demands.

Our study shows that a mixed solution where both virtual an physical CDN nodes are used can dramatically reduce the overall costs sustained by the operator to purchase and operate the distribution infrastructure. In particular, we observed that gains can be up to 65% when considering the cheapest vCDN price. Our contribution is also to formulate efficient solution algorithms for the two-stage stochastic planning problem that can scale to realistic topology sizes. Rather than solving the deterministic equivalent problem in the extensive form, our proposed L-shaped algorithm and heuristic can efficiently find a solution, saving up to 95% of time compared to the MILP solver.

Part IV

Security in NDN

CHAPTER 7

Confidentiality, Trackability and Access Policy Evolution

The NDN specification forces content producers to sign each Data packet, in order to guarantee content *integrity*, *provenance* and *relevance* (see Sec. 2.2). It also suggests to enforce *confidentiality* by encrypting the content itself, and by providing the corresponding decryption keys only to the legitimate consumers.

However, the presence of universal in-network caching poses new issues related to the evolution of the *access policy*, and producers may even lose control on the data they provide. In fact, in NDN content producers must face novel difficulties: 1. it becomes very complex to revoke access privileges once that the decryption keys have been provided to the legitimate users and the content has been cached; 2. whenever intermediate caches directly respond to the consumers' requests, producers do not receive any access feedback and, therefore, they can hardly keep track of content accesses. Both these issues arise for the fact that intermediate caches can persist (and serve) the content even if it is encrypted with a key that was later revoked.

For all these reasons, this chapter tacles three important security requirements, namely *confidentiality*, *trackability* and *access policy evolution*.

Our contribution is to propose *ConfTrack-CCN*, an efficient encryption-based extension to the CCN/NDN proposal, designed to enforce *confidential* data dissemination, *trackable* content access and seamlessly support *policy evolution*, in a *cache-aware* manner. *ConfTrack-CCN* jointly enforces all these three requirements by protecting the data with two layers of encryption, the latter of which evolves to reflect access privilege updates. A forced consumer-producer interaction makes consumers fetch keying materials, while sending back logging data on the accessed objects.

We evaluate *ConfTrack-CCN* by performing thorough simulation campaign with real network topologies. The results clearly show that, on average, *ConfTrack-CCN* ensures a 20% higher hit-rate than other security schemes, while introducing a negligible computational overhead.

7.1 Introduction

NDN promises to boost the content distribution capabilities of the network by providing universal in-network caching as a default feature implemented right inside the network protocol stack [117, 194]. For this reason, this paradigm appears to be very appealing for producers selling digital contents in online stores, since NDN promises to drammatically cut their infrastructure costs accountable to content distribution.

A common type of contract that these services offer is the subscription plan: the customer pays for a periodic subscription which gives her/him the right to access a given subset of the content catalog for the whole duration of the purchased period. As soon as the right to access the catalog expires, the user should be denied any further access to the available collection of files. Moreover, to improve content recommendation systems, these services have great interests in having detailled statistics on which contents a user has been accessing.

However, due to the presence of universal in-network caching, in NDN any network node can potentially serve the requested content without sending any feedback to the origin server. Therefore, in current proposals for NDN, a consumer whose access privilege to a popular content has already been revoked might still fetch that data directly from the caches, and, worst of all, without even making the original producer be aware that the access violation occurred.

For all these reasons, in this chapter we propose *ConfTrack-CCN*, a novel cache-aware and encryption-based mechanism tailored for Named-Data Networking, designed to enforce *confidentiality*, *content access trackability* and support *access policy evolution* at the same time. Data is protected by two layers of symmetric encryption, the latter of which evolves to match access privilege updates, thus ensuring *confidentiality*. Moreover, we force the consumers to retrieve keying material directly from the original producers, an interaction that provides access *trackability* feedback. Lastly, we efficiently enforce *access policy evolution* by making the nodes refresh only a small fraction of the cached objects, a strategy that reduces the operational costs of network operators, since it leads to higher overall hit-rates when compared to those obtained with custom security solutions implemented at the application level. Our design is settled on top of the solid ground of standard, state of the art, symmetric encryption and hash functions, in order to leverage the efficient hardware implementations of these primitives [23].

In summary, we provide the following contributions:

— We propose *ConfTrack-CCN*, a novel security layer that enforces confidential and trackable content dissemination in a cache-friendly manner, while providing seamless support to the evolution of access policies.

— We perform a thorough analysis of the security of our proposal, by showing that

the mechanism ensures confidentiality when both the policy evolves and the access privilege of a user is revoked. We further provide evidence that it can also be used to detect and protect against malicious users that cooperate forming a coalition.

— We measure the performance of our proposed architecture by a simulation campaign in real network topologies. Numerical results show that *ConfTrack-CCN* always performs better than user-based encryption schemes, leading to a higher overall hit-rate, while introducing a negligible performance penalty.

— We analyze the cryptographic overhead of our proposal by implementing a Java prototype of *ConfTrack-CCN*. Collected results show that computationally demanding encryption procedures are executed only once on the content producer side, i.e., when a new content is published on the network; a negligible performance overhead is instead required on the consumer side.

This chapter is structured as follows: Sec. 7.2 presents the *user-based encryption* scheme and shows that it jeopardizes the benefits of in-network caching. Sec. 7.3 describes *ConfTrack-CCN* in detail, while in Sec. 7.4 we perform the security analysis of our proposal, even in the presence of users' collusion. A thorough performance evaluation is presented in Sec. 7.5. Sec. 7.6 discusses related works, and finally, Sec. 7.7 concludes the chapter.

7.2 User-based Encryption Scheme

A first scheme for enforcing all the security requirements is what we call *user-based encryption*. Despite the fact that one such scheme may have desirable security properties, it jeopardizes the efficiency of in-network caching by 1. duplicating the same objects and therefore increasing the size of the content catalog, and 2. disrupting the overall content popularity.

Let $\mathcal{U}$ be the set of consumers, $U \subseteq \mathcal{U}$ denotes a subset of users, while $\mathcal{O}$ is the set of objects published by a content provider. We define a *user-based* encryption scheme as follows: the content provider generates a new encryption key K_o^U, for each content $o \in \mathcal{O}$ that a given subset of users $U \subseteq \mathcal{U}$ is entitled to access.

By restricting the cardinality of users, ideally setting $|U| = 1$, many different copies of the same content (encrypted with different keys) will be published in the network. This feature not only makes the cardinality of the real object catalog processed by intermediate routers rise to the very large $|\mathcal{U}| \cdot |\mathcal{O}|$ size, but it also completely disrupts the overall object popularity statistics, severely affecting the efficiency of the caching infrastructure. However, one such scheme has interesting security properties: *confidentiality* is enforced by making the producer disclose the decryption keys only to the authorized users. Instead, *trackability* and *access policy evolution* are jointly supported by the fact that a very low cache hit-rate will be experienced and the producer can easily re-encrypt the content and publish new versions that fully enforce the new access policy.

On the other hand, user-based encryption can also be used as a mechanism that generates a high hit rate, but can only provide weak protection. In particular, one such

Table 7.1: *Summary of the notation used to illustrate the confidentiality mechanism.*

Summary of the Notation	
$\mathcal{B}$	The set of content blocks
$\mathcal{C}$	The set of chunks
n	The number of content blocks, $n = \|\mathcal{B}\|$
m	The number of chunks per content block, $m = \|\mathcal{C}\|$
K_b	The First-Layer encryption key used for content block $b \in \mathcal{B}$
s	The First-Layer encryption key size expressed in bits
$r_i, \forall i \in \{2, ..., m\}$	Sequences of s random bits
$\mathbb{K}$	Keying materials appended to the first chunk. $\mathbb{K} = K_b \oplus r_2 \oplus r_3 \oplus ... \oplus r_m$, where $\oplus$ is the bitwise XOR
$SK(h, l)$	The Second-Layer encryption key
$stp(h, l)$	Producer state of the key regression algorithm
$stc(h, l)$	Consumer state of the key regression algorithm
h	The version of the key
l	The seed used to initialize key regression

configuration can be achieved by selecting large $|U|$ values, for instance using only one encryption key per object (i.e.: $U = \mathcal{U}$).

While having nice security properties *user-based encryption* totally nullifies the content distribution benefits promised by NDN. In the next sections we will describe *ConfTrack-CCN* our security mechanism and show that it correctly implements the security requirements of *confidentiality*, *trackability* and *access policy evolution*, while outperforming user-based encryption in terms of network efficiency.

7.3 Design of ConfTrack-CCN

We now introduce *ConfTrack-CCN*, the novel mechanism we propose to enforce *confidential and trackable* content dissemination in CCN/NDN, thus representing a fundamental layer of the Named-Data Networking paradigm. Our vision is based on the following principles:

— **Encryption enforces confidentiality**. The producer provides encrypted content to the network. Caches store content without the right to access the plaintext.

— **Key management enforces trackability**. Consumers authenticate and directly retrieve decryption keys from the original publisher.

— **Access policy evolution is based on key-derivation and re-encryption**. The evolution of the access policy is enforced by the producer issuing a new version of the content, encrypted with a new key. There exists a key derivation mechanism to let the consumer compute keys for previous versions of a chunk.

In order to enforce the above-mentioned requirements, a producer encrypts the content *twice*, as detailed in the following two subsections: the First Layer of encryption forces consumers to retrieve all content chunks (Sec. 7.3.1), whereas the Second Layer ensures that only authorized users can retrieve the current version of keying material

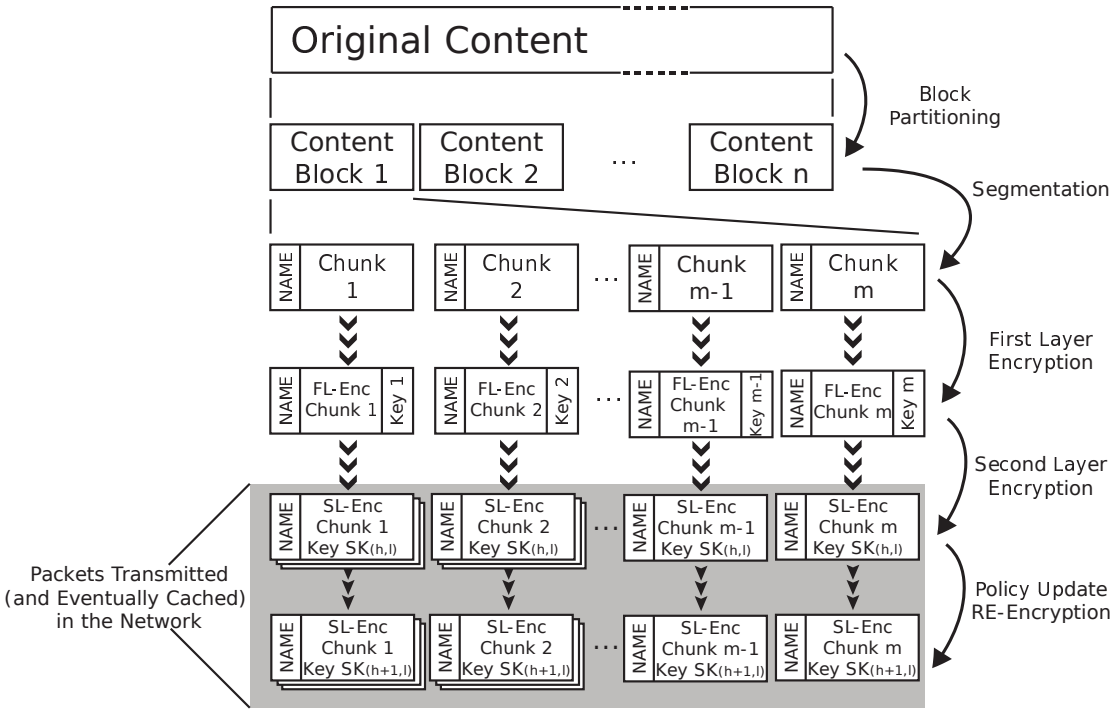


Figure 7.1: *High-level packet structure defined by* ConfTrack-CCN. *FL-Enc and SL-Enc denote the* 1^{st} *and* 2^{nd}*-layer encrypted chunks, respectively.*

(Sec. 7.3.2). In Sec. 7.3.3 we describe the authenticated key-retrieval protocol used to exchange encryption keys securely while sending content access trackability feedback. Finally, in Sec. 7.3.4 we show how policy evolution is seamlessly enforced by *ConfTrack-CCN*.

For the sake of clarity, throughout this section we use the notation summarized in Table 7.1. Moreover, to keep the presentation as clear as possible, we make the assumption that the particular application considered does not have stringent delay requirements, although *ConfTrack-CCN* can also be used for live-streaming (by setting a small size for the content block).

7.3.1 First Layer of Encryption

The purpose of the First Layer of encryption is to force the consumer to retrieve *all* the content blocks before having access to the plaintext version of the requested data. In order to publish the content, the producer will 1. partition and segment the content, 2. encrypt the data and 3. piggyback keying material as depicted in Fig. 7.1 and further described below.

Content partitioning and segmentation. The original content, whose size is usually quite large [1], is initially partitioned into a set of *content blocks* denoted with $\mathcal{B}$, of cardinality $n = |\mathcal{B}|$. Each content block $b \in \mathcal{B}$ is of fixed size (e.g., 1 Mbyte) and is segmented according to the CCN specification into a set of *chunks* denoted with $\mathcal{C}$,

1. We focus our description on data with size of many Mbytes, that well represents multimedia content, like videos.

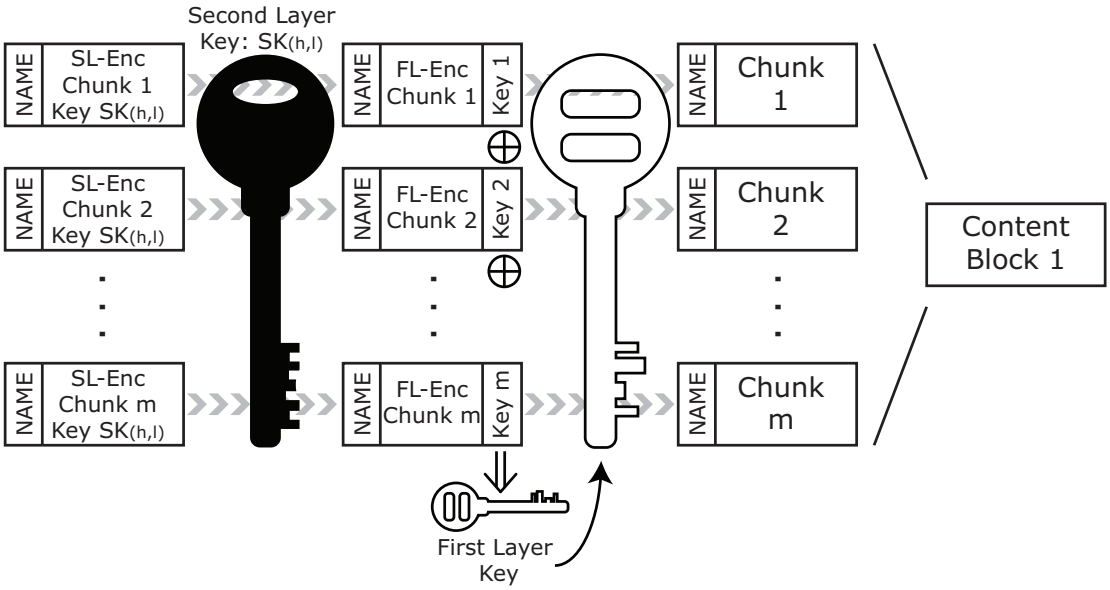


Figure 7.2: *The plaintext can be accessed only if the user can decrypt all the SL-Enc Chunks belonging to the content block, by using the corresponding Second Layer keys. Once that the FL-Enc Chunks are obtained, by computing the XOR of the pseudo-random values piggybacked to the FL-Enc Chunks, the user can compute the First Layer Key, which is then used to decrypt all the chunks belonging to the content block.*

significantly smaller than content blocks (e.g., 1 kbyte each). Each chunk will be transmitted through the network with a CCN data packet, and for this reason a CCN name is associated to each $c \in \mathcal{C}$. We denote with $m = |\mathcal{C}|$ the number of segmented packets per content block, which is computed as a function of the chosen chunk size (e.g., $m = 1000$).

Content encryption. As shown in Fig. 7.1, the First-Layer encrypted chunks, denoted by *FL-Enc Chunks*, are produced by encrypting the segmented *chunks*. The same *First-Layer encryption key*, K_b, is used to encrypt all the chunks belonging to the same content block $b \in \mathcal{B}$. Such key is randomly generated by the content producer, and is stored in a secure way in the FL-Enc chunks as described hereafter. Well-known symmetric encryption algorithms, such as AES in Cipher-Block Chaining (CBC) mode with PKCS#7 padding, can be used to secure the First Layer of encryption; in addition to that, we assume that the chosen key size is $s = 128$ bits (note that longer keys can be seamlessly used). We would like to point out the fact that only Second Layer encrypted chunks will be transmitted, and eventually cached, into the network.

Piggybacking keying material. As shown in Fig. 7.2, we force the consumers to retrieve all the encrypted packets in order to gain access to the plaintext version of the content block. We reserve the last s bits of FL-Enc chunks to transfer keying material.

The producer generates $m - 1$ random sequences of s bits $r_2, r_3, ..., r_m$ and then uses classical secret splitting techniques to compute:

— $K_b \oplus r_2 \oplus r_3 \oplus ... \oplus r_m = \mathbb{K}$, which is appended to the first chunk;

— $r_2, r_3, ..., r_m$, which are appended to the other chunks.

The joint utilization of CBC and the random values forces the client to retrieve all the encrypted packets in order to gain access to the plaintext. On the other hand, to implement access control and deny access to the entire content block to a given user, we make sure that he will not be able to decrypt at least one SL-Enc chunk, by not disclosing the corresponding Second Layer key. If the user cannot compute at least one FL-Enc chunk in the content block, he will not be able to compute the First Layer key K_b, and therefore, he will not be able to access the plaintext version of the content block.

7.3.2 Second Layer of Encryption

The purpose of the Second Layer of encryption is to guarantee confidentiality and prevent collusion, while serving as a basis to ensure trackability.

As depicted again in Fig. 7.1, after having been processed with the First Layer of encryption, chunks are further encrypted using one of the available Second-Layer encryption keys. Such keys are generated using the "Key Regression" derivation algorithm in its KR-RSA formulation [78]. We denote each Second-Layer key with $SK(h, l)$, where h is the key version, whereas l represents the original seed used to initialize the KR-RSA algorithm. A given FL-Enc chunk can be encrypted using many Second-Layer keys, each of which is generated using a different seed l.

As illustrated in Fig. 7.3, four operations (setup, wind key, unwind key, key derivation) define the KR-RSA algorithm. The detailed specification of all these operations is available in [78]. The initial *producer state*, $stp(0, l)$, is computed by the producer using the *setup* operation, whereas the *wind key* is used to derive $stp(h + 1, l)$ given $stp(h, l)$. The consumer retrieves the consumer state $stc(h, l)$ from the producer, by using the *authenticated key-retrieval protocol* described in Sec. 7.3.3. Given $stc(h, l)$, the consumer can therefore compute $SK(h, l)$ and all the previous versions $SK(h', l), 0 \leq h' < h$ by using the *unwind key* and the *key derivation* operations. We underline that, compared to well known key derivation algorithms (such as S/KEY [88]), KR-RSA provides two advantages: 1. the *wind key* operation can be executed an unbounded number of times, and 2. it is KR-secure, meaning that if two derived keys are released, a third party cannot discern whether they are linked or not.

Protection against collusion can be provided by means of encrypting the same FL-Enc chunk with many Second-Layer keys generated using different seeds: in this way, by disclosing different key sets to the users, we can detect malicious nodes that disclose their secret decryption keys even in the presence of collusion, as discussed in Sec. 7.4.2.

7.3.3 Authenticated Key-Retrieval Protocol

Hereafter, we describe the authenticated key-retrieval protocol used by the consumer to fetch the state $stc(h, l)$ from the producer, while sending content access trackability feedback. The data exchanged during such interaction comprises: 1. the ID of the object that the customer is willing to retrieve, $objID$; 2. the seed used to initialize KR-RSA, l, and 3. the version of the needed key, h.

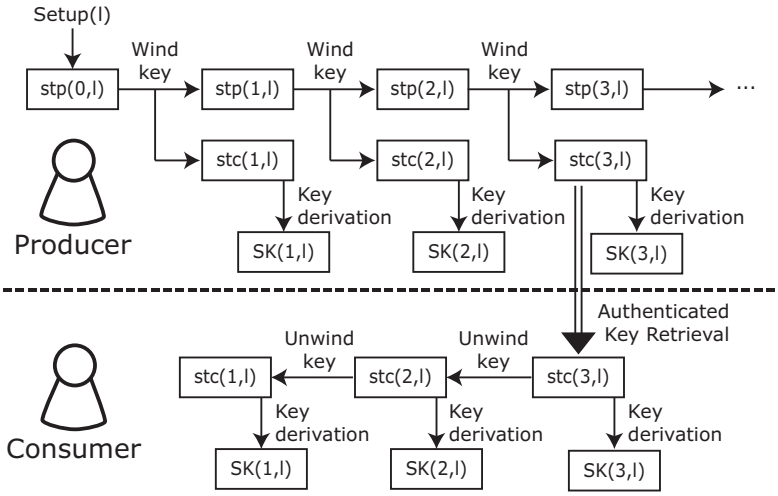


Figure 7.3: Key Regression - RSA*: $stp(h,l)$ denotes the producer state at version h initialized with seed l, whereas $stc(h,l)$ is the consumer state. The* wind key *operation lets the producer compute a new encryption key $SK(h+1,l)$, whereas the* unwind key *operation is used by the consumer to compute previous key versions.*

We assume that the producer is reachable through the globally routable name `/prod.com/`, and that he generated the asymmetric key-pair $KeyPair_p = (Pub(p), Priv(p))$, whereas the consumer publishes content under the name `/cons.com/` and owns the keys $KeyPair_c = (Pub(c), Priv(c))$. We denote with $||$ the concatenation operator, with E_k(data) the encryption of "data" with key k; finally, the hash of "data" is represented by $hash(data)$.

Fig. 7.4 summarizes the sequence of messages exchanged. MSG_1 is used to communicate to the content producer the namespace `/cons.com/rnd1` under which authentication data was published. MSG_1 is at the same time confidential and authentic, since it is encrypted with $Pub(p)$, and signed with $Priv(c)$. MSG_2 is used to retrieve key information and to communicate the random number (rnd2) that will be used by the customer to generate MSG_4 and retrieve the Second-Layer encryption key. MSG_3 contains the basic information `/objID/blockID/chunkID/seedID`, whereas MSG_4 and MSG_5 complete the protocol by permitting to obtain the Second-Layer encryption key, if the consumer is authorized to fetch it.

As shown in Fig. 7.4, cache hits can only be generated while retrieving SL-Enc chunks. This is due to the presence of timestamps in the naming structure we designed for the authenticated key retrieval protocol. Both the consumer and producer send CCN interest packets during the key retrieval dialog. Since the CCN model is pull driven, this type of dialog is necessary to let one host upload data to a remote server. In particular, the host will publish on the network the content to be uploaded while the server will issue interest packets to retrieve the corresponding data.

The amount of data that nodes exchange in order to transmit keying materials is practically negligible: less than 5 kbytes must be transmitted for each content block. The overhead introduced in the *Authenticated Key-Retrieval Protocol* for a content block size of 1 Mbyte (as considered in our numerical results) is therefore less than 1%.

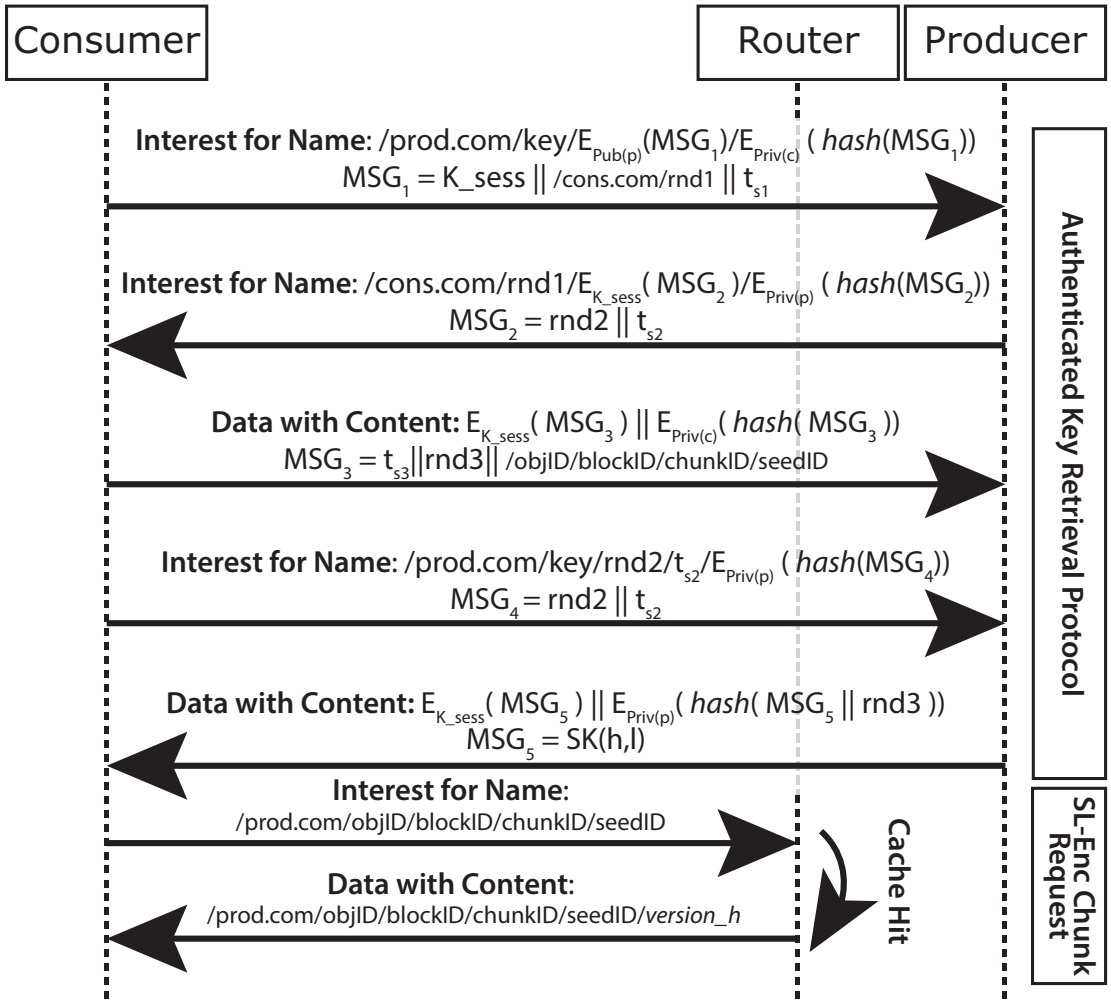


Figure 7.4: *Messages exchanged in the* Authenticated Key-Retrieval Protocol *when the consumer downloads the Second-Layer encryption key, while jointly authenticating and providing access trackability feedback.*

7.3.4 Policy Evolution

ConfTrack-CCN lets the producer make the access policy evolve by revoking access to the customers not entitled anymore to make use of given resources. In order to enforce policy evolution, the producer: (1) computes new encryption keys $SK(h+1,l)$; (2) re-encrypts and publishes the new version of the data; (3) does not serve anymore the obsolete content and (4) does not disclose the new $stc(h+1,l)$ to the customers whose access privilege was revoked. More specifically, the rationale behind this approach is to make sure that some chunks of a *content block* will be served to the revoked user encrypted with the $SK(h+1,l)$ key, as enforced by (1)-(3). In this way, the user cannot obtain the corresponding decryption key $stc(h+1,l)$, as enforced by (4), and consequently he will not be able to decrypt the corresponding chunk. Finally, even though the user can retrieve all the chunks in one content block, he will still not be able to decrypt the whole content block.

Hereafter, we describe the policy evolution mechanism we designed, in order to

make the Content-Centric Network quickly refresh contents as to ensure enforcement of confidentiality properties.

Even if caches can still provide obsolete content as a result of an incoming interest, the cache evolution mechanism we design is tailored to support such evolution in an efficient manner. To achieve this goal, we propose an innovative caching policy mechanism that, at regular intervals, forces the cache to forward an upstream interest for a "fresh" copy of a single chunk randomly chosen in a given content block. We implement such behavior by making the router mark cached content as *stale*, and forcing it to evict the data from the cache. Our mechanism and the joint presence of the two layers of encryption ensure that the network will propagate policy evolution changes very quickly, minimizing the amount of data that nodes exchange. It is important to note that our proposal is backward-compatible: caches not willing to utilize our proposal will work as usual, but they will experience higher operational costs due to lower hit rates.

We suggest to utilize the naming scheme shown hereafter:
`/prod.com/objID/blockID/chunkID/seedID/`h. The consumer issues interests without specifying h, the version of the Second-Layer encryption key: by issuing an interest for
`/prod.com/objID/blockID/chunkID/seedID/` the consumer is demanding any version of the given content chunk, a condition that ensures higher hit rates.

When receiving an interest packet, caches execute Alg. 11.

Algorithm 11: Cache Update Algorithm

Input : prod.com, objID, blockID, chunkID, seedID
Output: DataPkt

```
1  if IsCached(prod.com, objID, blockID, chunkID, seedID) then
2      t ⇐ GetTimeout(prod.com, objID, blockID);
3      if IsExpired(t) then
4          c ⇐ ChooseCachedRandomChunk(prod.com, objID, blockID);
5          MarkChunkAsStale(c);
6          UpdateTimeout(prod.com, objID, blockID);
       end
7      DataPkt ⇐ GetDataPkt(prod.com, objID, blockID, chunkID);
   else
8      ForwardInterestUpstream(prod.com, objID, blockID, chunkID);
   end
```

If the interest generates a cache-hit, the "freshness" timeout for
`/prod.com/objID/blockID/chunkID` is checked (Steps 1-3). If such timeout is expired, the cache will select a random chunk already cached by the node and belonging to the same content block; such chunk will be refreshed by making the router mark the content as stale (Steps 4-5), while the timeout will be updated (Step 6). Finally, the data packet will then be provided to the customer, by reading it from the content store in case of a cache hit (Step 7), or by forwarding the interest upstream (Step 8).

7.4 Security Analysis

In this section we analyze *ConfTrack-CCN* by studying the security properties of the proposed mechanism. In particular, in Sec. 7.4.1 we prove that an adversary whose access privilege has been revoked cannot access the plaintext version of the content, if at least one SL-Enc chunk has been refreshed. In Sec. 7.4.2 we instead investigate the problem of user collusion, and show that ConfTrack-CCN provides strong security properties even in the presence of a coalition of adversaries.

7.4.1 Cache Management Security

In this section we prove that, as a result of an access policy evolution for a given content, our proposed scheme revokes access to a user who is not authorized to obtain the given content anymore. The rationale behind such property is to make sure that the user cannot decrypt *at least one* SL-Enc chunk of a given content block b. If this is the case, our mechanism guarantees that the user cannot access the plaintext of the *entire* content block, since he cannot obtain the First Layer decryption key. Hereafter we consider the scenario where the access privilege of an adversary $\mathbb{A}$ has been revoked and the caching policy mechanism proposed in Sec. 7.3.4 has refreshed at least one SL-Enc chunk of the block b.

Theorem 6. *The first encryption layer forces the users to retrieve all the FL-Enc chunks belonging to the same content block b to gain access to the plaintext version of the data.*

Proof. In order to prove the security of *ConfTrack-CCN*, we take into account the most adverse case where $\mathbb{A}$ could retrieve m SL-Enc chunks, but he was able to decrypt only $m-1$ of them; hence, $\mathbb{A}$ can access all the FL-Enc chunks of a block apart from the j-th. Since the FL-Enc chunks contain

$$\begin{cases} r_j & \text{if } j \neq 1 \\ K_b \oplus r_2 \oplus r_3 \oplus ... \oplus r_m = \mathbb{K} & \text{if } j = 1, \end{cases} \tag{7.1}$$

both cases must be considered in the proof.

Case $j \neq 1$: the adversary can access $\mathbb{K}$ and all the random values apart from the j-th (r_i, $\forall i : i \neq j$). Thus, he can compute $\mathbb{K} \oplus r_2 \oplus r_3 \oplus ... \oplus r_{j-1} \oplus r_{j+1} \oplus ... \oplus r_m = K_b \oplus r_j$. This cyphertext is the one obtained using the Vernam cipher, an information-theoretically secure encryption algorithm that provides perfect secrecy under the assumption that the sequence $r_2, r_3, ..., r_m$ is truly random and will be used only once. Therefore $\mathbb{A}$ cannot compute the decryption key K_b.

Case $j = 1$: $\mathbb{A}$ can access all the FL-Enc chunks, apart from the first. For this reason, he doesn't know the value of $\mathbb{K}$, but he can extract the sequence of random values $r_2, r_3, ..., r_m$. However, these random values are uncorrelated with the encryption key K_b, which cannot be computed by the adversary. □

As a result of the access policy evolution, *ConfTrack-CCN* can prevent unauthorized accesses to the resources distributed in the network. Our mechanism only necessitates

to make sure that at least one SL-Enc chunk cannot be decrypted by the revoked user. In this way, as Theorem 6 shows, access to the *entire* content block is denied.

7.4.2 Detecting Collusion Attacks

As we will show in Sec. 7.5, *ConfTrack-CCN* uses shared encryption keys to increase the overall hit-rate that the network can achieve while reducing also the computational overhead of the solution. Hereafter, we investigate the problem of user collusion aiming at disclosing the Second Layer decryption keys to let unauthorized users decrypt the content that the provider has published in the network. We will show that, by distributing different key sets to the users, *ConfTrack-CCN* can also provide *collusion prevention* and *detection* functionalities.

We do not take into account the scenario where an adversary $\mathbb{A}$ provides the plaintext version of the content, because, due to the large data size, we assume that it is cost ineffective for him to serve the plaintext. On the other hand, it is instead feasible for $\mathbb{A}$ to distribute his own set of decryption keys, due to their modest size.

The content provider may want to distribute the content using only one set of Second Layer keys, shared among all the users. In this case the overall hit-rate would be very high, since only one copy of the content is going to be distributed in the network, but the security of the mechanism would be very low since an adversary $\mathbb{A}$ can easily disclose the set of keys without revealing his own identity. On the other hand, the provider may want to use different sets of Second Layer encryption keys, making sure that each user has his own unique set. In this case, if an adversary discloses his keys, the content provider can easily detect his identity and therefore he will be able to punish the user for this unauthorized behavior. In this subsection, we consider a scenario in between these two extremes: the content provider assigns different key sets to the users; therefore, when performing a key disclosure, the user is implicitly disclosing information on his own identity. In our model, the content provider should be able to identify the misbehaving user, given the disclosed keys, even in the challenging scenario where users collaborate and form a coalition.

With the ILP model presented hereafter we compute the best key disclosure strategy that the coalition of adversaries can choose. Their objective is to minimize the maximum identity disclosure of the users in the coalition, in order to make it difficult for the content provider to identify them as guilty. The key assignment strategy implemented by the content provider to allocate the Second Layer decryption keys has an important impact on the overall collusion-resistance properties of the proposed system. To perform one such analysis, we take into account the adverse scenario where the provider randomly allocates the Second Layer keys to its users. As reported in Fig. 7.5, even in this unfavorable case, *ConfTrack-CCN* exhibits strong security properties, detecting users' collusion with very high probability.

For the sake of clarity, Table 7.2 summarizes the notation used in this section. Let $\mathcal{M}$ be the set of malicious users, $\mathcal{N}$ the set of non-malicious users, while $\mathcal{U} = \mathcal{M} \cup \mathcal{N}$ is the set of all the user we consider in our analysis. We denote with $\mathcal{B}$ the set of content blocks, while $\mathcal{K}$ is the set of encryption keys.

We consider the most adverse case where malicious users collude. We make the re-

Table 7.2: *Summary of the notation used for the Collusion Analysis.*

Parameters of the model	
$\mathcal{M}$	Set of malicious users
$\mathcal{N}$	Set of non-malicious users
$\mathcal{U}$	Set of all users ($\mathcal{U} = \mathcal{M} \cup \mathcal{N}$)
$\mathcal{B}$	Set of content blocks
$\mathcal{K}$	Set of encryption keys
$y_{u,b,k}$	Key assignment matrix. $y_{u,b,k} = 1$ if user $u \in \mathcal{U}$ is assigned for block $b \in \mathcal{B}$ the key $k \in \mathcal{K}$; otherwise, $y_{u,b,k} = 0$.

Decision Variables of the Model	
$x_{u,b,k}$	0-1 Variable that indicates if the identity of user $u \in \mathcal{U}$ is disclosed for block $b \in \mathcal{B}$ with respect to the encryption key $k \in \mathcal{K}$
d^M	Identity disclosure for malicious users
d^N	Identity disclosure for non-malicious users

alistic assumption that colluding members do not know which keys have been assigned to non-malicious users; hence, their best key disclosure strategy is to minimize the maximum identity exposure of every member in the coalition by publishing a *combination* of the decryption keys possessed by them.

We denote with $y_{u,b,k}$ the key assignment matrix: $y_{u,b,k} = 1$ if key k is given to user u for content b, while $y_{u,b,k} = 0$ otherwise. $x_{u,b,k}$ is a binary decision variable we use to model both the set of disclosed keys and user identities. In particular, if key $k \in \mathcal{K}$ for content $b \in \mathcal{B}$ is disclosed by some malicious user, all the users $u \in \mathcal{U}$ for which $y_{u,b,k} = 1$ will implicitly disclose their identities (and thus, their $x_{u,b,k}$ variable will be set to 1). In other terms, while disclosing a key for a given block, all the other users to whom that key was assigned will implicitly disclose their identities at the same time.

We denote with d^M the maximum number of times the identity of a *malicious user* is disclosed by the mechanism. On the other hand, d^N represents the maximum number of times a *non-malicious user* is deemed to be guilty. We say that the mechanism is collusion-resistant if the content provider can identify at least one colluding user as guilty of disclosing his secret decryption keys. In other words, the maximum number of times a malicious user is thought to be guilty by the content provider should be greater than the number of times the content provider thinks another user is responsible for disclosing the key sequence. Therefore, if $d^M > d^N$, we say that the mechanism is collusion resistant.

The optimal key disclosure strategy for $\mathbb{A}$ is formulated as an ILP model as follows:

$$\min d^M \tag{7.2}$$

subject to:

$$\sum_{\forall p \in \mathcal{U}} x_{p,b,k} \leq (x_{u,b,k} + 1 - y_{u,b,k}) \, |\mathcal{U}| \qquad \forall (u,b,k) \in \mathcal{U} \times \mathcal{B} \times \mathcal{K} \quad (7.3)$$

$$\sum_{\substack{\forall m \in \mathcal{M} \\ \forall k \in \mathcal{K}}} x_{m,b,k} \geq 1 \qquad \forall b \in \mathcal{B} \quad (7.4)$$

$$x_{u,b,k} \leq y_{u,b,k} \qquad \forall (u,b,k) \in \mathcal{U} \times \mathcal{B} \times \mathcal{K} \quad (7.5)$$

$$\sum_{\substack{\forall b \in \mathcal{B} \\ \forall k \in \mathcal{K}}} x_{m,b,k} \leq d^M \qquad \forall m \in \mathcal{M} \quad (7.6)$$

$$x_{u,b,k} \in \{0,1\} \qquad \forall (u,b,k) \in \mathcal{U} \times \mathcal{B} \times \mathcal{K}. \quad (7.7)$$

The objective function (7.2) finds the best key disclosure strategy that malicious users can adopt. Since they do not know which encryption keys are possessed by non-malicious users, they will therefore minimize their maximum identity disclosure. This explains why d^N does not appear in the objective function.

The set of constraints (7.3) ensures that if a user in the coalition discloses his key, all the other users possessing the same key will disclose their identities as well.

In (7.4), we ensure that at least one key will be disclosed for each block, whereas the set of coherence constraints (7.5) forces a user to disclose only the keys he possesses.

Constraints (7.6) force d^M to represent the maximum identity exposure of a user. We add the binary constraints on the disclosure variable in (7.7).

Lastly we compute the maximum number of times the identity of non-malicious users is disclosed, d^N, as follows:

$$d^N = \max_{\forall n \in N} \left(\sum_{\substack{\forall b \in \mathcal{B} \\ \forall k \in \mathcal{K}}} x_{n,b,k} \right). \quad (7.8)$$

To evaluate the collusion resistance property of *ConfTrack-CCN*, we consider different scenarios with 20 or 25 encryption keys and up to 2000 blocks. We set the total number of users equal to 50, we uniformly generate the key assignment matrix $y_{u,b,k}$, and we perform the analysis with up to 80% colluding members. For each of these scenarios we consider 100 random collusion sets, then, using the model (7.3)-(7.7) and equation (7.8), we compute the *detection probability*. This latter is defined as the ratio between the number of scenarios where the content provider identifies a guilty user in the coalition and the total number of scenarios considered. Fig. 7.5 shows the detection probability as a function of the size of the colluding users set.

As expected, since encryption keys are uniformly allocated to the users, increasing the number of keys and/or chunks is beneficial in terms of resilience against users' collusion. In particular, 25 keys and 2000 blocks are sufficient to detect a coalition containing up to 80% of colluding users, while if we decrease the number of keys to 20, we can detect misbehaving consumers with very high probability even when 60% of the users are colluding.

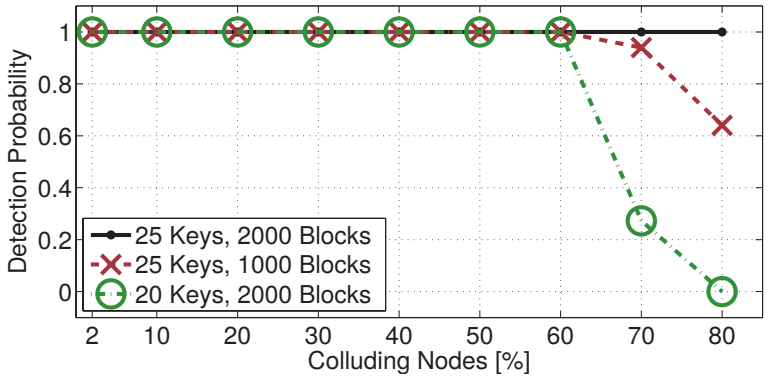


Figure 7.5: Disclosure Detection Analysis*: detection probability as a function of the size of the collusion set, for 20 or 25 encryption keys and up to 2000 blocks. The total number of users is set to 50.*

In summary, our proposed mechanism proves to be very robust against collusion, even in adverse case when the content provider randomly distributes the keys to the users, and several of them are colluding.

7.5 Numerical Results

In this section we provide extensive numerical evaluations of our proposal through simulations as well as using the Java prototype we built. More specifically, Sec. 7.5.1 quantifies the *bandwidth benefits* in terms of cache hit-rate that our proposal can provide with respect to the alternative scenario where a *user-based* encryption model is used. In Sec. 7.5.2 we study the security properties of our solution, whereas Sec. 7.5.3 provides computational performance measurements on the prototype of *ConfTrack-CCN*.

7.5.1 Cache Hit Analysis

We first quantify the bandwidth benefits that our proposal can guarantee using an analytic, as well as a simulated model.

We denote with $\mathcal{U}$ the set of consumers, whereas $\mathcal{O}$ denotes the set of objects. For the sake of simplicity, we assume that objects have the same size Θ; however, our results can be easily extended to the general case where objects have a variable size. Each consumer $u \in \mathcal{U}$ generates requests modeled as a Poisson process with aggregate rate λ_u, proportional to the object popularity p_o, where $o \in \mathcal{O}$.

We assume that the popularity distribution of objects is subject to the *Independent Reference Model* (IRM), since this leads to very realistic results when the requests are generated by a large population of users, as shown in [76]. Object popularity is a *Zipf* distribution.

We take into account the *least recently used* (LRU) cache replacement policy [46]. Under these assumptions, our performance analysis is based on the model proposed in [46], as it was shown in [76] that this is an extremely accurate model for hierarchical caches. According to such model, the hit rate for object $j \in \mathcal{O}$ is given by $h(j) =$

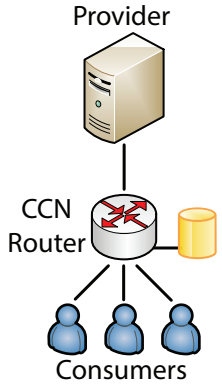


Figure 7.6: *Single-level cache network topology used for numerical results.*

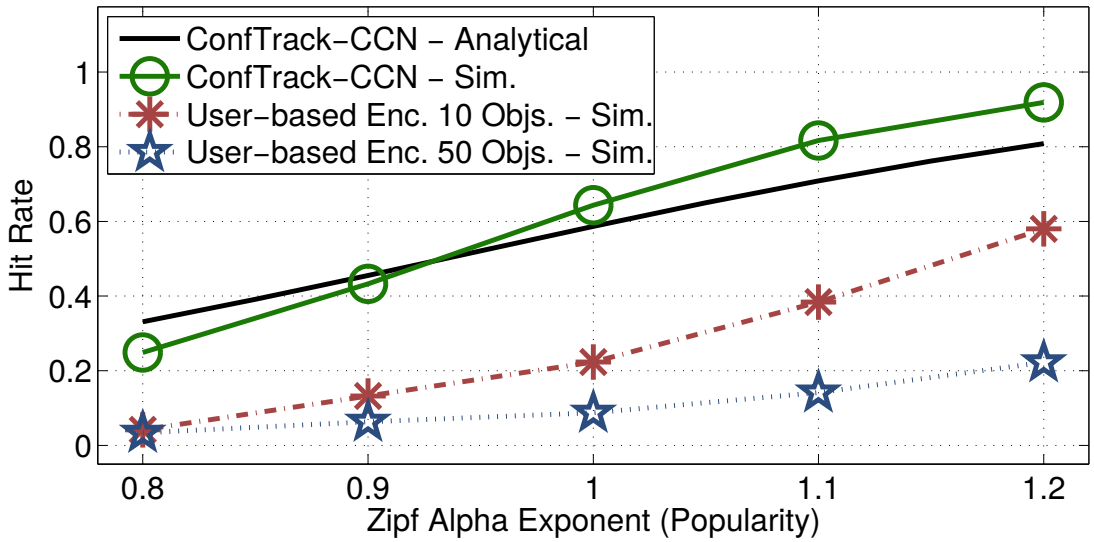


Figure 7.7: *Hit-rate for a single-level LRU cache network with* 10^8 *objects of 1 Mbyte each and 100 Mbyte of storage, as a function of the Zipf popularity exponent* α.

$1 - e^{-p_j t_C}$, where t_C is the root of $\sum_{j=1}^{|\mathcal{O}|} (1 - e^{-p_j t_C}) = M$, M being the cache size, expressed in number of objects.

We compute the hit rate for a single-level cache network (shown in Fig. 7.6) for $\alpha \in [0.8; 1.2]$, because this range of popularity exponents well represents heterogeneous types of contents, as discussed in [155]. Fig. 7.7 illustrates the analytic solution of the model, as well as simulation results obtained using the ndnSIM network simulator [15] for the single-level cache network with 10^8 objects of 1 Mbyte each, given a 100 Mbyte cache. In particular, we consider the following cases:

1. *ConfTrack-CCN* is used, and 0.1% of each object is encrypted with 5 different keys;
2. User-based encryption is used (as described in Sec. 7.2), and each object is accessed by 10 users;
3. User-based encryption is used, and each object is accessed by 50 users.

As shown in Fig. 7.7, the hit rate obtained by *ConfTrack-CCN* is more than 20%

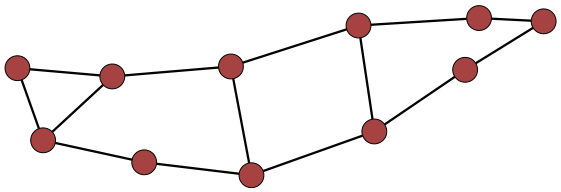


Figure 7.8: *Abilene topology used for numerical results.*

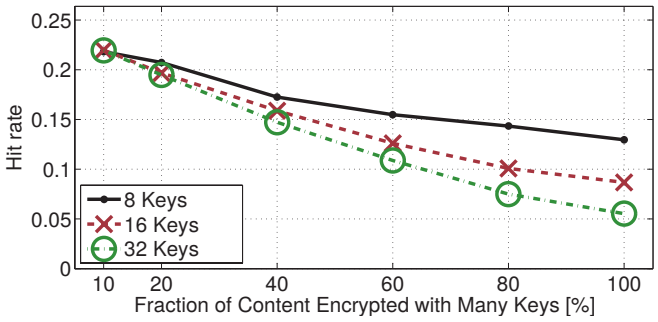


Figure 7.9: *Hit-rate measured in the Abilene topology as a function of the percentage of contents encrypted with many keys.*

higher than the one provided when user-based encryption is used, even when few users (i.e., 10) are accessing the object set.

We further investigated the hit rate considering the Abilene network topology (depicted in Fig. 7.8) [155], with 10^8 objects of 10 Mbytes each, a Zipf exponent $\alpha = 1.2$, a caching storage of 1 Gbyte per node, and varying the following parameters:

1. The number of encryption keys used;
2. The fraction of objects encrypted with many keys;
3. The policy evolution delay.

As illustrated in Fig. 7.9, a higher hit rate is obtained by reducing the number of Second-Layer encryption keys used, as well as the fraction of objects encrypted with many of these keys. This result is expected, since there is a clear trade-off between the performance of the network and the security of the mechanism: if we increase the number of encryption keys, we increase also the possibility to detect colluding users.

Fig. 7.10 plots the hit rate as a function of the policy evolution delay. When we impose very frequent policy updates (e.g., once every 0.1 seconds), the network hit rate drops below 5%, whereas if the content is updated every 10 seconds or above (a more realistic assumption), we instead observe that a hit rate higher than 25% is obtained when 8 keys are used.

Similar results, omitted here due to space constraints, have been observed also in other network topologies, including the Géant network [7].

7.5.2 Security Analysis

In this section we numerically study the security properties enforced by our proposed *ConfTrack-CCN* solution, comparing it with a scenario where no other security

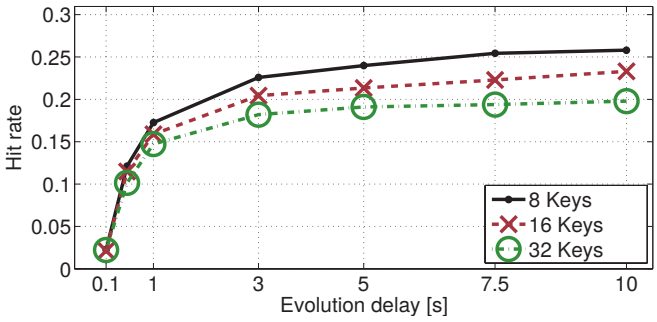
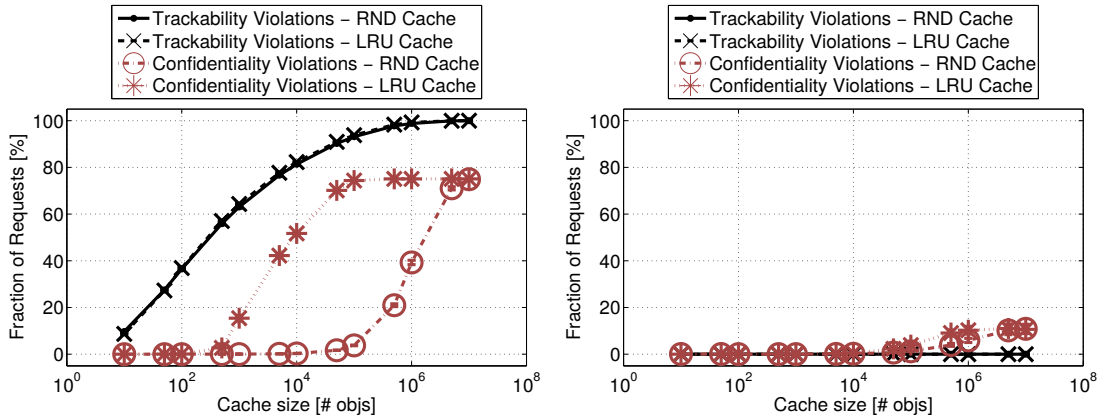


Figure 7.10: *Hit-rate measured in the Abilene topology as a function of the policy evolution delay.*



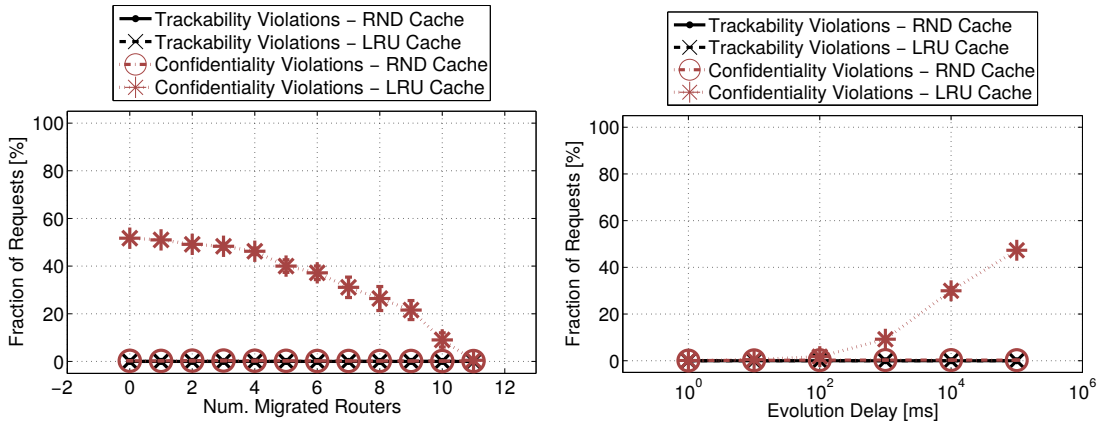
(a) ConfTrack-CCN is not-used **(b)** ConfTrack-CCN is used

Figure 7.11: Security of ConfTrack-CCN - Effect of cache size. *Fig. 7.11a shows the trend of trackability and confidentiality violations as a function of the cache size, when ConfTrack-CCN is* not *utilized. Fig. 7.11b shows the trend for the same security properties, when ConfTrack-CCN is used and all network routers implement our proposed policy evolution mechanism with a policy evolution delay of 1 second.*

mechanism is implemented in the network.

We make the realistic assumption that different traffic classes are handled by the network; in particular, we consider a traffic mix as in [76] where Video on Demand (VoD) and User-Generated Content (UGC) are delivered. As detailed in [76], we assume that UGC traffic is characterized by a content catalog of 10^8 objects whose average size is 10 MB, the Zipf exponent is $\alpha = 0.8$ and accounts for 38% of the overall amount of requests. On the other hand, VoD traffic is characterized by a catalog of 10^4 objects of 100 MB each, $\alpha = 1.2$, and accounts for the remaining 62% of the requests. We simulate such scenario in the Abilene topology, as described in the previous section, and study the security properties of our proposed mechanism where CCN routers implement caching functionalities using either the LRU or a random cache replacement policy. We assume that the content provider is offering a VoD service, and it wants to securely distribute the content into the network.

In order to provide evidences that *ConfTrack-CCN* successfully satisfies the secu-

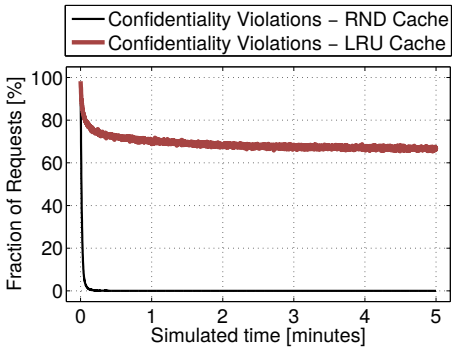


(a) Effect of the Number of Migrated Routers **(b)** Effect of the Cache Evolution Delay

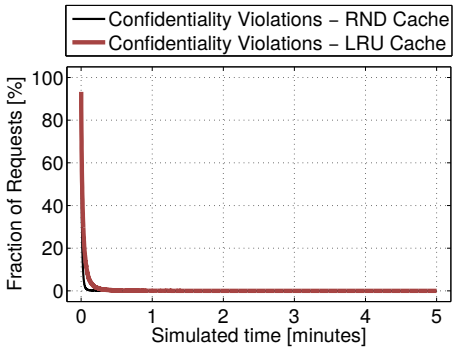
Figure 7.12: Security of ConfTrack-CCN - Effect of Number of Migrated Routers and Cache Evolution Delay. *Both Fig. 7.12a and 7.12b show the security properties of a network where ConfTrack-CCN is used. In Fig. 7.12a we portray the effect of the number of routers implementing our novel cache eviction policy, while in Fig. 7.12b we show the effect of the evolution delay parameter for the cache eviction policy.*

rity requirements, we measure security violations, as defined hereafter. In particular, a *"trackability violation"* arises whenever a consumer can acquire a given content without making the provider be aware that he did so. We assume that users do not disclose their decryption keys, therefore *"confidentiality violations"* happen only in the case of *access policy evolution*. For this reason, to simulate a scenario where the access policy changes, we assume that the producer removes the 100 most popular resources he is publishing, and we measure the number of successful requests the consumers can issue on the subset of these deleted contents. For each of the analysis we performed 20 different runs, and figures portray the very narrow 95% confidence intervals. Unless stated otherwise, we consider a cache size of 10^4 objects, all the routers implement our proposed policy evolution mechanism, and a policy evolution delay of 1 second is used.

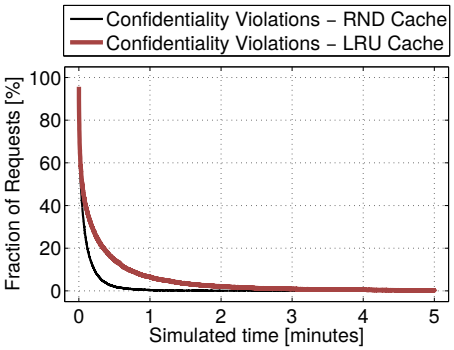
In Fig. 7.11a we show the security properties of a network that does not implement any protection mechanism, as a function of the cache size. We observe that, in terms of *trackability violations*, random caching and LRU show practically the same performance; in particular, even considering small cache sizes (in the order of 10^3 objects), more than 60% of requests are served without making the producer be aware that users accessed the content. This behavior is caused by the high efficiency of the caching mechanism, perfectly in line with the results observed in [76]. On the other hand, as shown in Fig. 7.11a, random cache replacement is to be preferred in terms of confidentiality violations. In particular, LRU caches tend to persist popular contents even if the provider has deleted them from its catalog. Fig. 7.11b reports instead the performance observed in the same scenario simulated for Fig. 7.11a, but when *ConfTrack-CCN* is used, and all the routers implement our proposed cache update mechanism. In this case, trackability violations do not happen anymore, whereas a small number of confidentiality violations ($< 18\%$) are observed when caches are very large and can store up to 10^7 objects ($1/10$ of the entire object catalog). For the sake of conciseness, we



(a) ConfTrack-CCN not-used, 10^4 objs. cached



(b) ConfTrack-CCN used, 10^4 objs. cached



(c) ConfTrack-CCN used, 10^5 objs. cached

Figure 7.13: Temporal Evolution of Confidentiality Violations. *We portray the number of confidentiality violations in time, for different types of cache eviction policies. In Fig. 7.13a we show the behavior of a standard CCN network that does not use ConfTrack-CCN, while in Fig. 7.13b we show the effect of using our solution. Lastly, in Fig. 7.13c we consider a cache of one order of magnitude larger, while still using ConfTrack-CCN.*

are not reporting here similar results for the User-based encryption since, as thoroughly described in the previous section, *ConfTrack-CCN* outperforms user-based encryption in terms of the overall network hit-rate.

The effect of the number of routers implementing our proposed cache update policy, and the effect of the evolution delay parameter (i.e.: the timeout of Alg. 11, Sec. 7.3.4) are reported in Fig. 7.12a and 7.12b, respectively. Since *ConfTrack-CCN* is used, no trackability violations are observed, moreover the number of confidentiality violations when using a random cache replacement policy is negligible regardless of both these parameters. LRU, instead, is sensitive to both the number of routers implementing our policy and the evolution delay. In particular, as Fig. 7.12a shows, by migrating many routers to our proposed cache update policy, we can significantly reduce the number of observed confidentiality violations, dropping from 55% of confidentiality violations (when no router implements our policy), down to almost 0% (when all the routers implement it). On top of that, as observed from Fig. 7.12b, the cache evolution delay has a remarkable effect on the observed number of confidentiality violations, and, as

expected, the lower the timeout, the more secure the network is.

By adopting our *ConfTrack-CCN* mechanism, trackability violations do not happen anymore, moreover the number of confidentiality violations is significantly reduced. Hereafter, we study the transient behavior of the network in the interval immediately following the instant when contents are removed from the producers, as a result of an access policy evolution. In particular, since our proposed cache update algorithm does not require nodes' cooperation, a small delay is introduced to effectively revoke the access to the deleted contents, as shown in Fig. 7.13.

Fig. 7.13a and 7.13b show the advantages of using *ConfTrack-CCN* and our proposed cache update mechanism. The trends confirm previous observations on the LRU cache replacement policy: in terms of *confidentiality violations*, one such replacement policy does not provide adequate protection against confidentiality violations. However, by adopting *ConfTrack-CCN* as well as our cache update mechanism, we can make the network quickly remove stale content making the number of confidentiality violations drop to almost 0% in less than 1 minute. Lastly, in Fig. 7.13c we study the effect of the cache size in a scenario where *ConfTrack-CCN* is still used. In particular, even by increasing the caching storage of one order of magnitude, the number of confidentiality violations is still kept under control, and quickly reaches 0% in less than 3 minutes.

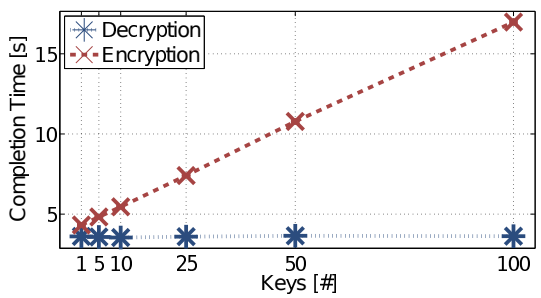
7.5.3 Prototype Encryption Performance

As described in Sec. 7.3, *ConfTrack-CCN* requires the end-hosts to implement cryptographic primitives, in particular the producers encrypt the content, whereas the consumers perform the corresponding decryption. Intermediate routers need not execute any of these cryptographic functions, and therefore this design choice ensures scalability of our security architecture, making it possible to operate at line speed. In this subsection, we quantify the computational overhead introduced on the end-hosts by *ConfTrack-CCN* to perform the cryptographic operations required to implement our mechanism.

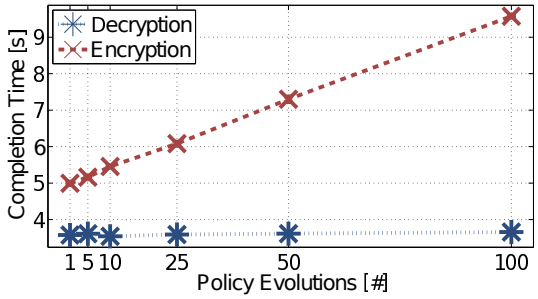
To perform such evaluation, we implemented a Java prototype of the encryption primitives of *ConfTrack-CCN*, and performed extensive evaluations to understand the performance impact of these procedures. All tests have been executed on a dual Intel Xeon 2.2GHz machine, with 64GB RAM, running Ubuntu Linux 12.04.2 LTS, using Java SE 7u25 with the Bouncy Castle provider for the Java Cryptography Extension and the Java Cryptography Architecture. The performance metric considered in all these evaluations is the completion time of the cryptographic primitives, as a function of:

1. The *number of encryption keys*;
2. The *number of policy updates*;
3. The *size of the data*, expressed in Mbytes.

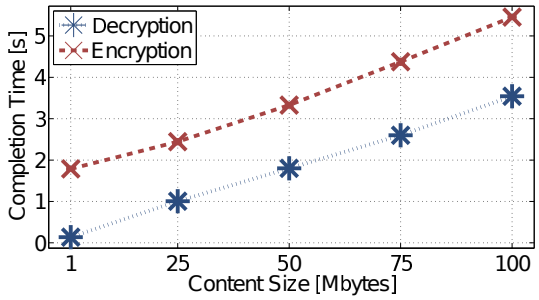
Unless otherwise stated, we performed the tests encrypting 100 Mbytes of data with 10 different keys, considering 10 policy updates. For each scenario that we took



(a) Effect of the number of keys



(b) Effect of the number of policy evolutions



(c) Effect of the content size

Figure 7.14: Encryption/Decryption Completion Time. *The figures represent the computational overhead introduced to handle the encryption / decryption of 100 Mbytes of data, with 10 keys assuming 10 policy updates (unless stated otherwise). Fig. 7.14a represents the effect of the number of keys. Fig. 7.14b shows the effect of the number of policy evolutions, whereas in Fig. 7.14c the effect of the content size is shown.*

into account we performed 10 runs, and in Fig. 7.14a-7.14c we further reported the corresponding (narrow) 95% confidence intervals.

As described in [78], the KR-RSA algorithm generates a public-private RSA key pair for each Second Layer encryption key. Despite the fact that such generation is computationally expensive, our proposed mechanism minimizes its performance impact in two ways: 1. the key-pairs have to be computed only once that new data is published on the network, and 2. the pairs can also be pre-computed offline.

When releasing a new version of the data by performing a policy update, modular exponentiation is used to *wind/unwind* the keys; the other operations require only to perform fast symmetric AES encryption. It is interesting to note that the fraction of re-encrypted content only affects the network hit-rate as discussed in Sec. 7.5.1, while it does not have any impact on the computational performance, since it only requires to choose an appropriate Second-Layer key among those already generated.

Fig. 7.14a shows the encryption/decryption completion time as a function of the number of Second-Layer keys. A linear time is required to compute the RSA key-pairs, and such computational cost is only paid on the content provider side. Furthermore, we observe that cryptographic primitives introduce only minor overhead on the consumer side.

The policy evolution frequency as well as the size of the data are positively correlated with the completion time of the algorithm, as depicted in Fig. 7.14b and 7.14c, respectively. The gap between the encryption/decryption curves is due to the cost for initializing the RSA key-pairs, which is paid only by the content provider, once the data is originally published on the network.

7.6 Related Work

A thorough discussion of the security properties of NDN has been provided in Chapter 2, Sec. 2.2, in particular Sec. 2.2.4 reviews recent works on access control in NDN. In this section we would like to compare our ConfTrack-CCN proposal with respect to recent literature.

Few proposals to enforce access control at the application level have already been formulated; for instance, in [201], Zhu et al. propose a mechanism to enforce confidentiality for a conference tool in NDN, while Burke et al. formulate in [31] a proposal to secure light control. Both these works are application-specific results, that can hardly be extended to other more general cases in which a content producer wants to securely distribute large content files through NDN.

Fotiu et al. study in [72] an access control enforcement delegation mechanism for an ICN, which guarantees consumers' privacy in the PURSUIT [73] architecture. However, one such proposal is not tailored for NDN; in fact, while in PURSUIT can be assumed that a caching node, called *Relaying Party* (RP), will provide the content only to the subset of authenticated users, one such requirement can instead hardly be enforced considering in-network caching as implemented in NDN. On top of that, their focus on privacy preservation makes it very difficult to jointly satisfy the *trackability* requirement.

Advanced public-key cryptosystems were used by different authors to enforce the common requirement of ensuring access control in ICN: Zhang et al. in [195] leverage Identity-Based Encryption (IBE); Proxy Re-Encryption (PRE) is instead used in [186], to support *access-control* in a content-centric network; Misra et al. in [133] used instead Broadcast Encryption (BE) and, more specifically, a public-key traitor tracing variant of Shamir's threshold secret sharing scheme.

To cope with the large files distribution, the solutions reviewed so far use one layer of symmetric encryption to efficiently secure the data, while enforcing access control protection by using different versions of public-key encryption (such as IBE, PRE, BE), to securely distribute keying materials, and ensuring that only authorized users can retrieve the correct decryption key. However, all these systems can be easily bypassed by making users disclose the common symmetric encryption key used to initially protect the content. On top of that, since the symmetric key is shared by all the users, one such strategy does not reveal any type of information regarding the user's identity. Our *ConfTrack-CCN* mechanism, instead, is tailored to avoid this issue. In fact, in our design, contents are encrypted twice, using two symmetric keys. To break the system security, users must disclose the Second Layer decryption keys, but, doing that, they are also forced to expose their identity. Furthermore, by relying on standard hash functions

Table 7.3: *Comparison of Related Works on Access Control.*

	ConfTrack-CCN	[72]	[195]	[186]	[133]	[115]
Confidentiality	✓	✓	✓	✓	✓	✓
Privacy Preservation	-	✓	-	-	-	-
Trackability	✓	✗	✗	-	✗	✗
Access Policy Evolution	✓	-	-	✓	✓	-
Collusion Protection	✓	-	-	✗	✓	✗
Cache Awareness	✓	In PURSUIT, not in CCN	✗	✓	✓	✓
Performance	Sym. Enc. Hash Functions	Sym. Enc.	IBE	PRE	BE	Cypher Indep-endent

and symmetric encryption, our solution significantly improves the overall efficiency of the mechanism.

In [115] Kurihara et al. present CCN-AC, an encryption-based framework for CCN to implement any access control scheme. By leveraging CCN 1.0 manifests, they show that CCN-AC supports group based, as well as broadcast access control. Their proposal does not directly deal with policy evolution, but they discuss the tradeoff between the speed of access revocation, with respect to the efficiency of the caching mechanism.

For the sake of clarity, Table 7.3 provides a thorough comparison of our ConfTrack-CCN scheme with respect to the most notable reviewed literature on access control enforcement.

7.7 Conclusion

We proposed *ConfTrack-CCN*, a *cache-aware, encryption-based* mechanism designed to enforce confidential and trackable content dissemination in Information-Centric Networks, while seamlessly supporting policy evolution. With our mechanism, the consumers can take advantage of the distributed caches (one of the core elements of NDN) to retrieve the encrypted data, while they are instead forced to contact directly the content producer to fetch keying material, thus authenticating themselves and providing access trackability feedback.

We evaluated our solution by both developing a mathematical model and through a simulation analysis with real network topologies, showing that the proposed mechanism always performs better than user-based encryption. In the most adverse case, *ConfTrack-CCN* ensures a 25% hit rate, while user-based encryption schemes can barely reach 5%. A large gap is also observable in the best case, where our solution scores a 91% hit-rate, compared to the 58% of user-based schemes. By distributing different key sets to the users, our proposal provides robust collusion detection and prevention functionalities, even when 60% of the users are colluding.

On top of that, we also quantified the computational overhead introduced by *ConfTrack-CCN* by implementing a Java prototype of our solution and performing extensive measurements. We demonstrated that on the consumer side our solution has a negligible performance cost since it only executes fast operations (it decrypts 100 Mbytes of data in less than 4 seconds). At the same time, our design limits the impact of demanding cryptographic primitives such as the generation of the Second-Layer encryption keys: this operation has to be performed on the content provider side only once.

CHAPTER 8

Conclusion

We begin this chapter by providing in Sec. 8.1 the overall conclusions of this Ph.D. thesis, whereas potential future works are thoroughly discussed in Sec. 8.2.

8.1 Summary of Contributions

In this thesis we investigated the general problem of efficient in-network content distribution, and we leveraged the novel paradigm of Information-Centric Networks, focusing on the design known as Named-Data Networking (NDN), to provide an effective response to one such need. In particular, after presenting in Part I the state of the art on the most relevant aspects concerning ICN, we provided novel contributions to three main topics: 1. wireless named-data networking (Part II); 2. network planning for content distribution (Part III) and 3. security in NDN (Part IV). As detailed hereafter, our contributions specifically addressed the efficiency property of in-network content distribution: we wanted our work to shed some lights on the evaluation of the benefits that the ICN architecture may lead to, especially when compared to the standard ways content distribution is nowadays handled.

Our proposal for wireless NDN (Chapter 3) was specifically tailored to reuse spare bandwidth and storage resources available at residential WiFi gateways in a heterogeneous network. In this context, we proposed an auction mechanism to remunerate wireless access point owners willing to lease their unexploited bandwidth and storage resources. We envisioned the scenario where economic incentives are provided by the content provider in order to extend his distribution network, offering broadband access to mobile customers, while jointly offloading his distribution infrastructure with caches.

Our allocation schemes ensures the *individual rationality* and *truthfulness* properties, since forces the AP owners to declare the real valuations for the offered resources. Not only we provided a polynomial-time heuristic that preserves individual rationality and truthfulness, but we also studied the distributed scenario in which the mobile clients autonomously connect to the available access points. We modeled this latter case as a congestion game, we proved that there exists a unique Nash equilibrium, and we provided lower bounds on the price of anarchy. Our study was motivated and supported by the fact that due to the decreasing cost of storage [75], as well as recent advances for smart home-gateways [42, 105], we believe that in the near future, most domestic wireless access-points will likely be equipped with caching solutions. As a matter of fact, to the best of our knowledge, this is a common technique already used to pre-load popular contents, distributed by on-demand video streaming providers. In this way the content provider not only can reach a larger number of mobile clients, but he can also reduce the overall energy expenditures and costs to run the distribution infrastructure, thanks to offloading.

While formulating novel network planning models for content distribution we took into account the different problems of 1. object placement; 2. request routing and 3. replica server placement. In this context, we provided different contributions, by beginning in Chapter 4 to describe the migration step to an ICN. More in detail, we formulated a single time slot optimization model to discover the overall economic benefits the operator should expect as a result of migrating to one such networking paradigm. We complemented our contribution by designing a greedy heuristic, and by showing that it can compute close to optimal solutions while saving a significant amount of computation time. We observed that, as expected, the object popularity distribution has a remarkable impact on the final solution; on top of that, a rather surprising result is that nodes migration prices have a limited effect on the overall costs, and the operator can experience significant benefits even considering a very large span of the CAPEX costs. While our goal in this work was mostly performing an economic analysis of the benefits of the nodes migration, we believe that ICN solutions will also be appealing for network operators not only to reduce the costs, but also to improve the users' quality of experience, lowering the latency and network congestion.

In Chapter 5 we extended the model presented in Chapter 4, by taking into account a time-evolving content popularity. Moreover, we focused our analysis on a performance comparison between our model for a content delivery network, with respect to the one of an Information Centric Network. The main objective of this work was to shed some lights on the important problem of understanding whether one of the two architectures may naturally lead to better performance, compared to the other. In order to do so, we cast the problem as a mixed integer linear optimization model that we used to study the performance bounds of these architectures. Our key findings suggested that both these distribution infrastructure can dramatically reduce the amount of traffic that nodes should exchange, however there are no clear evidences that ICN should be preferred over CDN. On the other hand, we clearly observed that whenever the speed at which content popularity evolves is very high, ICN can accommodate better performance than those experienced in CDN.

Lastly, in Chapter 6 we took into account a virtualized content distribution infras-

tructure based on the novel paradigm of network functions virtualization. In our view, in one such networking architecture, physical CDN nodes will be backed by virtual surrogates implemented in software and running in the datacenter. Among the main advantages that a vCDN architecture can achieve compared to a physical-only infrastructure is the fact that it can natively cope with sudden traffic demand changes. As a matter of fact, new instances of virtualized CDN surrogates can easily be activated on demand, and they can also be placed in vantage points where they can foster dramatic traffic reductions. In order to take into account the probabilistic nature of traffic requests, we formulated the problem as a stochastic planning model in which the network operator performs a "here-and-now" decision given the uncertainty associated in the stochastic nature of the traffic demands. We solved this problem considering 1. the deterministic-equivalent extensive-form MILP formulation, 2. the L-Shaped decomposition, and 3. a polynomial-time greedy heuristic.

Security implications involved in ICN were taken into account in Chapter 7 where we tackled the important requirements of access-control and trackability, in the presence of distributed caches. While ICN solutions promise to improve security by forcing content producers to sign the data they publish on the network, the presence of distributed caches, however, makes it far from easy to revoke access to a user. In fact, distributed caches do not enforce any access-control and, upon a request for an object that they have in cache, they will always reply to the incoming request. On top of that, the content producer looses control on the data it publishes on the network, in fact, whenever a content chunk is directly served by an intermediate node as a result of a cache hit, the provider will be unable to receive any content access feedback on the given request, and it will be unaware of that access. In order to mitigate both these issues we formulated a security mechanism that we named ConfTrack-CCN, that leverages standard symmetric encryption and hash functions to enforce these security requirements.

8.2 Future Works

Research in Information Centric Networks gained momentum in recent years, and, as reviewed in Chapter 2, many problems have already received a lot of attention from the scientific community. However, as detailed in this section, we believe that the following major topics will likely lead to the most compelling achievements: 1. virtualized architectures for content delivery, and 2. optimal content placement and caching policies.

Virtualized architectures for content delivery. In Chapter 6 we presented our contribution considering a virtual content delivery architecture. While virtual networks are, generally speaking, a hot topic nowadays, some contributions specifically related to content delivery networks have recently begun to appear [108, 135, 151, 159]. However, it is still unclear what is the real relationship between these two technologies: a first attempt to fill this gap is provided in [151], where the authors consider both software and hardware programmability and virtualization, to jointly support, on the same network devices, heterogeneous instances of Information-Centric Networks. Nonethe-

less, an open problem is to characterize the performance differences between a vCDN architecture built on top of network functions virtualization, with respect to the Future Internet approach fostered by ICN designs.

In continuity with our contribution, we think that a promising research path would be to take into account the effect of dynamically changing the number of virtual CDN surrogates, with respect to the load on the origin and the other CDN nodes [29]. As a matter of fact, some components of the CDN architectures, such as the CDN request-routing module, will be impacted by changes to the number of virtual surrogate servers available, both when this number increases, as well as whenever it decreases. However, the most critical effects will certainly be experienced on the origin node, and the impact will strongly depend on whether the CDN is operated according to the origin-pull or the origin-push model [106]. Another potential extension is to consider a more accurate model for the traffic requests, that takes also into consideration their spatial correlation [85].

Optimal Content Placement and Caching Policies. The object placement problem is usually solved offline by adopting a centralized view of the network, and it is used to compute the best strategy to spread copies of the objects in the available caching storage. On the other hand, caching policies are usually conceived as distributed algorithms implemented by each cache node, and, in their collaborative version, they may even entail the possibility to exchange data between the caches as to introduce some form of coordination. Many caching policies have been formulated in recent years [40, 66, 152, 157, 172, 188]. While each of these contribution improves the caching performance with respect to a specific scenario or metric, what is often ignored is the computational cost (and for the collaborative approaches also the communication cost) that must be paid in order to implement these policies in practical scenarios.

For similar reasons, due to the outstanding number of the policies formulated in the literature, a promising research path would be to provide a unified benchmark framework to compare all these contributions. Having one such benchmark data would provide immediate benefits: network operators interested in understanding the performance of the proposed algorithms would be able to quickly compare the different solutions provided, moreover this data set would become useful also to evaluate future research proposals. To evidence the urgency of having one such unified view on the caching problem, consider that in most of the cases the performance of the caching algorithm is measured in terms of the hit-rate it leads to. Although this approach is very common in the literature, it was often criticized, and many authors proposed other metrics to evaluate the efficiency of a caching mechanism [40, 182].

Optimal content placement models, like those we discussed in Part III of this thesis can often outperform the localized caching policies, since they select network-wide the location of the content replicas that optimize the expected objective function. Although there are many accurate models to compute the average hit-rate of a cache [69, 79, 131], we still miss a theoretical bound on the approximation ratio of this local solution, compared to the global optimum. Interesting results were presented in [170], but they hold only for the special class of regular topologies where all nodes have the same number of neighbors.

Bibliography

[1] Amazon CloudFront Pricing. Last visited in Aug. 2015. URL: http://aws.amazon.com/cloudfront/pricing/.

[2] Amazon Kindle 3G Connectivity. Last visited in Aug. 2015. URL: http://www.amazon.com/gp/help/customer/display.html?nodeId=200505540.

[3] CCNx Applications & Use Cases. Last visited in Aug. 2015. URL: https://www.ccnx.org/applications-use-cases/.

[4] CCNX Community Meeting (CCNxCon 2013) Keynote - Palo Alto, CA, USA, Sep. 2013. Speaker: Glenn Edens. Last visited in Aug. 2015. URL: http://www.ccnx.org/events/ccnxcon-2013/.

[5] CCNx Website. Last visited in Aug. 2015. URL: http://www.ccnx.org/.

[6] Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update 2014–2019 White Paper. Last visited in Aug. 2015. URL: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white_paper_c11-520862.html.

[7] Géant Network Website. Last visited in Aug. 2015. URL: http://geant3.archive.geant.net/Network/NetworkTopology/pages/home.aspx.

[8] Named-Data Networking Tools and Applications. Last visited in Aug. 2015. URL: http://named-data.net/codebase/applications/.

[9] NDN Packet Format Specification. Version 0.2-alpha-2. Last visited in Aug. 2015. URL: http://named-data.net/doc/ndn-tlv/.

[10] The Internet Topology Zoo. Last visited in Aug. 2015. URL: http://www.topology-zoo.org/.

[11] Gergely Acs, Mauro Conti, Paolo Gasti, Cesar Ghali, and Gene Tsudik. Cache privacy in named-data networking. In *International Conference on Distributed*

Computing Systems (ICDCS), pages 41–51, Philadelphia, Pennsylvania, USA, July 2013. doi:10.1109/ICDCS.2013.12.

[12] Vijay Kumar Adhikari, Yang Guo, Fang Hao, Matteo Varvello, Volker Hilt, Moritz Steiner, and Zhi-li Zhang. Unreeling Netflix: Understanding and Improving Multi-CDN Movie Delivery. In *IEEE INFOCOM*, pages 1620–1628, Orlando, FL, USA, March 2012. IEEE. doi:10.1109/INFCOM.2012.6195531.

[13] Alexander Afanasyev, Jeffrey Burke, Lixia Zhang, Kc Claffy, Lan Wang, Van Jacobson, Patrick Crowley, Christos Papadopoulos, and Beichuan Zhang. Named Data Networking. *ACM SIGCOMM Computer Communication Review*, 44(3):66–73, 2014. doi:10.1145/2656877.2656887.

[14] Alexander Afanasyev, Priya Mahadevan, Ilya Moiseenko, Ersin Uzun, and Lixia Zhang. Interest Flooding Attack and Countermeasures in Named Data Networking. In *IFIP Networking*, pages 26–31, Brooklyn, New York, USA, May 2013. IEEE. URL: http://lasr.cs.ucla.edu/afanasyev/data/files/Afanasyev/ifip-interest-flooding-ndn.pdf.

[15] Alexander Afanasyev, Ilya Moiseenko, and Lixia Zhang. ndnSIM: NDN simulator for NS-3. Technical report, NDN, October 2012. URL: http://named-data.net/techreport/TR005-ndnsim.pdf.

[16] Alexander Afanasyev, Junxiao Shi, Lan Wang, Beichuan Zhang, and Lixia Zhang. Packet Fragmentation in NDN: Why NDN Uses Hop-By-Hop Fragmentation. Technical report, NDN, May 2015.

[17] Ayşegül Altin, Bernard Fortz, Mikkel Thorup, and Hakan Ümit. Intra-domain traffic engineering with shortest path routing protocols. *Annals of Operations Research*, 7(4):301–335, 2013. doi:10.1007/s10288-009-0113-0.

[18] Marica Amadeo, Claudia Campolo, Antonella Molinaro, and Giuseppe Ruggeri. Content-centric wireless networking: A survey. *Computer Networks*, 72:1–13, 2014. doi:10.1016/j.comnet.2014.07.003.

[19] David Applegate, Aaron Archer, Vijay Gopalakrishnan, Seungjoon Lee, and K. K. Ramakrishnan. Optimal content placement for a large-scale VoD system. In *Co-NEXT*, pages 1–12, Philadelphia, Pennsylvania, USA, December 2010. ACM. doi:10.1145/1921168.1921174.

[20] Alper Atamturk and Muhong Zhang. Two-Stage Robust Network Flow and Design Under Demand Uncertainty. *Operations Research*, 55(4):662–673, 2007. doi:10.1287/opre.1070.0428.

[21] Arjun Attam and Ilya Moiseenko. NDNBlue : NDN over Bluetooth. Technical report, NDN, November 2013.

[22] Walid Ben-Ameur and Eric Gourdin. Internet Routing and Related Topology Issues. *SIAM Journal on Discrete Mathematics*, 17(1):18–49, 2003. doi:10.1137/S0895480100377428.

[23] Ryad Benadjila, Olivier Billet, Shay Gueron, and Matt J B Robshaw. The intel AES instructions set and the SHA-3 candidates. *Lecture Notes in*

Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 5912 LNCS:162–178, 2009. doi:10.1007/978-3-642-10366-7_10.

[24] John R. Birge and François Louveaux. *Introduction to stochastic programming*. Springer Series in Operations Research and Financial Engineering. Springer, 2011. doi:10.1007/978-1-4614-0237-4.

[25] Andreas Bley. Approximability of unsplittable shortest path routing problems. *Networks*, 54(1):23–46, 2009. doi:10.1002/net.20303.

[26] Pierre Bonami, Lorenz T. Biegler, Andrew R. Conn, Gérard Cornuéjols, Ignacio E. Grossmann, Carl D. Laird, Jon Lee, Andrea Lodi, François Margot, Nicolas Sawaya, and Andreas Wächter. An algorithmic framework for convex mixed integer nonlinear programs. *Discrete Optimization*, 5(2):186–204, 2008. doi:10.1016/j.disopt.2006.10.011.

[27] Youmna Borghol, Siddharth Mitra, Sebastien Ardon, Niklas Carlsson, Derek Eager, and Anirban Mahanti. Characterizing and modelling popularity of user-generated videos. *Performance Evaluation*, 68(11):1037–1055, 2011. doi:10.1016/j.peva.2011.07.008.

[28] Juan Felipe Botero, Xavier Hesselbach, Michael Duelli, Daniel Schlosser, Andreas Fischer, and Hermann De Meer. Energy efficient virtual network embedding. *IEEE Communications Letters*, 16(5):756–759, 2012. doi:10.1109/LCOMM.2012.030912.120082.

[29] Niels Bouten, Jeroen Famaey, Rashid Mijumbi, Bram Naudts, Joan Serrat, Steven Latre, and Filip De Turck. Towards NFV-based multimedia delivery. In *IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pages 738–741, Ottawa, ON, Canada, May 2015. IEEE. doi:10.1109/INM.2015.7140364.

[30] Lee Breslau, Pei Cao, Li Fan, Graham Phillips, and Scott Shenker. Web caching and Zipf-like distributions: evidence and implications. In *IEEE INFOCOM*, volume 1, pages 126–134, New York, March 1999. IEEE. doi:10.1109/INFCOM.1999.749260.

[31] Jeff Burke, Paolo Gasti, Naveen Nathan, and Gene Tsudik. Securing Instrumented Environments over Content-Centric Networking: the Case of Lighting Control. In *IEEE INFOCOM Workshops*, pages 394–398, Turin, Italy, April 2013. IEEE. arXiv:1208.1336.

[32] Dariusz Bursztynowski, Mateusz Dzida, Tomasz Janaszka, Adam Dubiel, and Michal Rowicki. HTTP/CCN Gateway and Cooperative Caching Demonstrator. Technical report, Sophia Antipolis, France, September 2012.

[33] Michael R. Bussieck and Stefan Vigerske. MINLP Solver Software. *Wiley Encyclopedia of Operations Research and Management Science*, pages 1–17, 2010.

[34] Byoung Hoon Jung, Nah-Oak Song, and Dan Keun Sung. A Network-Assisted User-Centric WiFi-Offloading Model for Maximizing Per-User Throughput in a Heterogeneous Network. *IEEE Transactions on Vehicular Technology*, 63(4):1940–1945, May 2014. doi:10.1109/TVT.2013.2286622.

[35] Giovanna Carofiglio, Massimo Gallo, and Luca Muscariello. ICP: Design and evaluation of an interest control protocol for content-centric networking. In *IEEE INFOCOM Workshops*, pages 304–309, Orlando, FL, USA, March 2012. IEEE. doi:10.1109/INFCOMW.2012.6193510.

[36] Giovanna Carofiglio, Massimo Gallo, and Luca Muscariello. Joint hop-by-hop and receiver-driven interest control protocol for content-centric networks. *ACM SIGCOMM Computer Communication Review*, 42(4):491, 2012. doi:10.1145/2377677.2377772.

[37] Giovanna Carofiglio, Massimo Gallo, Luca Muscariello, and Michele Papalini. Multipath Congestion Control in Content-Centric Networks. In *IEEE INFOCOM Workshops*, pages 3403–3408, Turin, Italy, April 2013. IEEE.

[38] Giovanna Carofiglio, Massimo Gallo, Luca Muscariello, and Diego Perino. Modeling data transfer in content-centric networking. In *International Teletraffic Congress (ITC)*, pages 111–118, San Francisco, CA, USA, September 2011.

[39] Giovanna Carofiglio, Vinicius Gehlen, and Diego Perino. Experimental evaluation of memory management in content-centric networking. In *IEEE International Conference on Communications (ICC)*, pages 1–6, Kyoto, Japan, June 2011. IEEE. doi:10.1109/icc.2011.5962739.

[40] Giovanna Carofiglio, Leonce Mekinda, and Luca Muscariello. LAC: Introducing Latency-Aware Caching in Information-Centric Networks. Technical report, June 2015. arXiv:1506.06642.

[41] Giovanna Carofiglio, Giacomo Morabito, Luca Muscariello, Ignacio Solis, and Matteo Varvello. From content delivery today to information centric networking. *Computer Networks*, 57(16):3116–3127, 2013. arXiv:arXiv:1202.0108v1, doi:10.1016/j.comnet.2013.07.002.

[42] Claudio Casetti, Yan Grunenberger, Frank Den Hartog, Anukool Lakhina, Henrik Lundgren, Marco Milanesio, Anna-kaisa Pietilainen, Shuang Zhang, and Renata Teixeira. Network Monitoring Architecture based on Home Gateways. In *Future Network and MobileSummit 2013*, pages 2–5, Lisbon, Portugal, 2013. IEEE. Last visited in Aug. 2015. URL: http://hal.upmc.fr/hal-00866819/document.

[43] Meeyoung Cha, Haewoon Kwak, Pablo Rodriguez, Yong-yeol Ahn, and Sue Moon. I tube, you tube, everybody tubes: analyzing the world's largest user generated content video system. In *ACM Internet Measurement Conference (IMC)*, pages 1–14, San Diego, CA, USA, October 2007. ACM. doi:10.1145/1298306.1298309.

[44] Wei Koong Chai, Diliang He, Ioannis Psaras, and George Pavlou. Cache "less for more" in information-centric networks (extended version). *Computer Communications*, 36(7):758–770, 2013. doi:10.1016/j.comcom.2013.01.007.

[45] Hung-Bin Chang and Kwang-cheng Chen. Auction-Based Spectrum Management of Cognitive Radio Networks. *IEEE Transactions on Vehicular Technology*, 59(4):1923–1935, 2010. doi:10.1109/TVT.2010.2042474.

[46] Hao Che, Ye Tung, and Zhijun Wang. Hierarchical web caching systems: Modeling, design and experimental results. *IEEE Journal on Selected Areas in Communications*, 20(7):1305–1314, 2002. doi:10.1109/JSAC.2002.801752.

[47] Yu-Chang Chen, Ja-Hsing Hsia, and Yi-Ju Liao. Advanced seamless vertical handoff architecture for WiMAX and WiFi heterogeneous networks with QoS guarantees. *Computer Communications*, 32(2):281–293, February 2009. doi:10.1016/j.comcom.2008.10.014.

[48] Xiang Cheng, Sen Su, Zhongbao Zhang, Hanchi Wang, Fangchun Yang, Yan Luo, and Jie Wang. Virtual network embedding through topology-aware node ranking. *ACM SIGCOMM Computer Communication Review*, 41(2):38, 2011. doi:10.1145/1971162.1971168.

[49] Raffaele Chiocchetti, Dario Rossi, and Giuseppe Rossini. CcnSim: An highly scalable CCN simulator. In *IEEE International Conference on Communications*, pages 2309–2314, Budapest, June 2013. IEEE. doi:10.1109/ICC.2013.6654874.

[50] Jaeyoung Choi, Jinyoung Han, Eunsang Cho, Ted Taekyoung Kwon, and Yanghee Choi. A survey on content-oriented networking for efficient content delivery. *IEEE Communications Magazine*, 49(3):121–127, 2011. doi:10.1109/MCOM.2011.5723809.

[51] N.M. Mosharaf Kabir Chowdhury and Raouf Boutaba. Network virtualization: state of the art and research challenges. *Communications Magazine, IEEE*, 47(7):20–26, 2009. doi:10.1109/MCOM.2009.5183468.

[52] N.M. Mosharaf Kabir Chowdhury, Muntasir Raihan Rahman, and Raouf Boutaba. Virtual Network Embedding with Coordinated Node and Link Mapping. In *IEEE INFOCOM*, pages 783–791, Rio de Janeiro, Brazil, April 2009. IEEE. doi:10.1109/INFCOM.2009.5061987.

[53] Alberto Compagno, Mauro Conti, Paolo Gasti, and Gene Tsudik. Poseidon: Mitigating interest flooding DDoS attacks in named data networking. In *Conference on Local Computer Networks, LCN*, pages 630–638, Sydney, Australia, October 2013. IEEE. arXiv:1303.4823, doi:10.1109/LCN.2013.6761300.

[54] Mauro Conti, Paolo Gasti, and Marco Teoli. A lightweight mechanism for detection of cache pollution attacks in Named Data Networking. *Computer Networks*, 57(16):3178–3191, 2013. doi:10.1016/j.comnet.2013.07.034.

[55] Eric Cronin, Sugih Jamin, Cheng Jin, Anthony R. Kurc, Danny Raz, and Yuval Shavitt. Constrained mirror placement on the Internet. *IEEE Journal on Selected Areas in Communications*, 20(7):1369–1382, 2002. doi:10.1109/JSAC.2002.802066.

[56] Jakub Czyz, Mark Allman, Scott Iekel-johnson, Eric Osterweil, and Michael Bailey. Measuring IPv6 Adoption. *ACM SIGCOMM Computer Communication Review*, 44(4):87–98, 2014. doi:10.1145/2619239.2626295.

[57] Jie Dai, Fangming Liu, Bo Li, Baochun Li, and Jiangchuan Liu. Collaborative caching in wireless video streaming through resource auctions.

IEEE Journal on Selected Areas in Communications, 30(2):458–466, 2012. doi:10.1109/JSAC.2012.120226.

[58] György Dán and Niklas Carlsson. Power-law Revisited : A Large Scale Measurement Study of P2P Content Popularity. In *9th International Workshop on Peer-to-peer Systems (IPTPS)*, pages 1–11, San Jose, California, USA, April 2010. IEEE.

[59] Christian Dannewitz. NetInf: An Information-Centric Design for the Future Internet. In *3rd GI/ITG KuVS Workshop on The Future Internet*, volume 1-3, pages 1–3, Munich, Germany, May 2009.

[60] Steven DiBenedetto and Paolo Gasti. ANDaNA : Anonymous Named Data Networking Application. In *Network and Distributed System Security Symposium, (NDSS)*, pages 1–20, San Diego, CA, USA, February 2012. The Internet Society. arXiv:1112.2205.

[61] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. In *Conference on USENIX Security Symposium SSYM*, volume 13, page 21, San Diego, CA, USA, August 2004. doi:10.1.1.4.6896.

[62] Soumitra Dixit, Shalini Periyalwar, and Halim Yanikomeroglu. Secondary user access in LTE architecture based on a base-station-centric framework with dynamic pricing. *IEEE Transactions on Vehicular Technology*, 62(1):284–296, 2013. doi:10.1109/TVT.2012.2221753.

[63] Andreas Eisenblätter and Jonas Schweiger. Multistage stochastic programming in strategic telecommunication network planning. *Computational Management Science*, 9(3):303–321, 2012. doi:10.1007/s10287-012-0143-5.

[64] Jocelyne Elias, Fabio Martignon, Antonio Capone, and Eitan Altman. Non-cooperative Spectrum Access In Cognitive Radio Networks: A Game Theoretical Model. *Computer Networks*, 55(17):3832–3846, 2011.

[65] ETSI. Network Functions Virtualisation - Update White Paper. Last visited in Aug. 2015. URL: http://portal.etsi.org/NFV/NFV_White_Paper2.pdf.

[66] Chao Fang, F. Richard Yu, Tao Huang, Jiang Liu, and Yunjie Liu. An energy-efficient distributed in-network caching scheme for green content-centric networks. *Computer Networks*, 78:119–129, February 2015. doi:10.1016/j.comnet.2014.09.017.

[67] Seyed Kaveh Fayazbakhsh, Yin Lin, Amin Tootoonchian, Ali Ghodsi, Teemu Koponen, Bruce M. Maggs, K. C. Ng, Vyas Sekar, and Scott Shenker. Less Pain, Most of the Gain: Incrementally Deployable ICN. *ACM SIGCOMM Computer Communication Review*, 43(4):147, 2013. doi:10.1145/2486001.2486023.

[68] Flavio Figueiredo, Fabrício Benevenuto, and Jussara M Almeida. The tube over time: characterizing popularity growth of youtube videos. In *4th ACM International Conference on Web search and Data Mining (WSDM)*, page 745, Hong Kong, China, February 2011. ACM. doi:10.1145/1935826.1935925.

[69] N C Fofack, P Nain, G Neglia, and D Towsley. Analysis of TTL-based cache networks. In *International Conference on Performance Evaluation Methodologies and Tools (VALUETOOLS)*, pages 1–10, Cargèse, France, October 2012. IEEE. doi:10.4108/valuetools.2012.250250.

[70] Bernard Fortz and Mikkel Thorup. Internet traffic engineering by optimizing OSPF weights. In *IEEE INFOCOM*, volume 2, pages 519–528, Tel Aviv, Israel, March 2000. IEEE. doi:10.1109/INFCOM.2000.832225.

[71] Nikos Fotiou, Somaya Arianfar, Mikko Särelä, and George C. Polyzos. A framework for privacy analysis of ICN architectures. In *Lecture Notes in Computer Science*, number October, pages 1–226. Springer, 2012. doi:10.1007/978-3-319-06749-0.

[72] Nikos Fotiou, Giannis F. Marias, and George C. Polyzos. Access control enforcement delegation for information-centric networking architectures. *ACM SIGCOMM Computer Communication Review*, 42(4):497, 2012. doi:10.1145/2377677.2377773.

[73] Nikos Fotiou, Pekka Nikander, Dirk Trossen, and George C. Polyzos. Developing information networking further: From PSIRP to PURSUIT. *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering*, 66 LNICST:1–13, 2012. doi:10.1007/978-3-642-30376-0_1.

[74] Nikos Fotiou, D. Trossen, Giannis F. Marias, A. Kostopoulos, and George C. Polyzos. Enhancing information lookup privacy through homomorphic encryption. *Security and Communication Networks*, 2:71–81, 2013. doi:10.1002/sec.

[75] Gary Francis. Data Storage – Trends and Directions. Technical report, Oracle Preservation and Archiving Special Interest Group (PASIG) EMEA Meeting, London, UK, 2011. Last visited in Aug. 2015. URL: http://lib.stanford.edu/files/pasig-jan2012/11B7FrancisPASIG_2011_Francis_final.pdf.

[76] Christine Fricker, Philippe Robert, and James Roberts. A versatile and accurate approximation for LRU cache performance. In *International Teletraffic Congress (ITC)*, volume abs/1202.3, pages 1–16, Krakow, Poland, September 2012. arXiv:arXiv:1202.3974v1.

[77] Christine Fricker, Philippe Robert, James Roberts, and Nada Sbihi. Impact of traffic mix on caching performance in a content-centric network. In *IEEE INFOCOM Workshops*, pages 310–315, Orlando, FL, USA, March 2012. IEEE. arXiv:arXiv:1202.0108v1, doi:10.1109/INFCOMW.2012.6193511.

[78] Kevin Fu, Seny Kamara, and Tadayoshi Kohno. Key regression: Enabling efficient key distribution for secure distributed storage. In *13th Annual Network & Distributed System Security Symposium (NDSS)*, San Diego, CA, USA, February 2006. URL: http://scholarworks.umass.edu/cgi/viewcontent.cgi?article=1147&context=cs_faculty_pubs.

[79] Massimo Gallo, Bruno Kauffmann, Luca Muscariello, Alain Simonian, and Christian Tanguy. Performance evaluation of the random replacement policy for networks of caches. *Performance Evaluation*, 72:16–36, 2014. arXiv:arXiv:1202.4880v1, doi:10.1016/j.peva.2013.10.004.

[80] J.J. Garcia-Luna-Aceves. Name-based content routing in information centric networks using distance information. In *1st International Conference on Information-Centric Networking - (ICN)*, pages 7–16, Paris, France, September 2014. ACM. doi:10.1145/2660129.2660141.

[81] Mehran Garmehi, Morteza Analoui, Mukaddim Pathan, and Rajkumar Buyya. An economic replica placement mechanism for streaming content distribution in Hybrid CDN-P2P networks. *Computer Communications*, 52:60–70, 2013. doi:10.1016/j.comcom.2014.06.007.

[82] Paolo Gasti, Gene Tsudik, Ersin Uzun, and Lixia Zhang. DoS and DDoS in named data networking. In *International Conference on Computer Communications and Networks, ICCCN*, pages 1–7, Nassau, Bahamas, July 2013. arXiv:arXiv:1208.0952, doi:10.1109/ICCCN.2013.6614127.

[83] Cesar Ghali, Gene Tsudik, and Ersin Uzun. Needle in a Haystack : Mitigating Content Poisoning in Named-Data Networking. In *NDSS Workshop on Security of Emerging Networking Technologies (SENT)*, San Diego, CA, USA, February 2014.

[84] Ali Ghodsi, Scott Shenker, Teemu Koponen, Ankit Singla, Barath Raghavan, and James Wilcox. Information-centric networking: Seeing the Forest for the Trees. In *ACM Workshop on Hot Topics in Networks /HotNets)*, pages 1–6, Cambridge, MA, USA, November 2011. ACM. doi:10.1145/2070562.2070563.

[85] Lazaros Gkatzikis, Vasilis Sourlas, Carlo Fischione, Iordanis Koutsopoulos, and György Dán. Clustered Content Replication for Hierarchical Content Delivery Networks. In *IEEE International Conference on Communications (ICC)*, page 6, London, UK, June 2015. IEEE.

[86] David Goergen, Thibault Cholez, and Thomas Engel. Security Monitoring for Content-Centric Networking. In *Data Privacy Management and Autonomous Spontaneous Security*, pages 274–286. 2013.

[87] Giulio Grassi, Davide Pesavento, Giovanni Pau, Rama Vuyyuru, Ryuji Wakikawa, and Lixia Zhang. VANET via named data networking. In *IEEE INFOCOM Workshops*, pages 410–415, Toronto, Canada, April 2014. IEEE. doi:10.1109/INFCOMW.2014.6849267.

[88] Neil M Haller. The S/KEY One-Time Password System. In *Symposium on Network and Distributed System Security*, pages 151–157, February 1994. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.44.6790&rep=rep1&type=pdf.

[89] Bing Han, Xiaofei Wang, Nakjung Choi, Ted "Taekyoung" Kwon, and Yanghee Choi. AMVS-NDN : Adaptive Mobile Video Streaming and Sharing in Wire-

less Named Data Networking. In *IEEE INFOCOM Workshops*, pages 375–380, Turin, Italy, April 2013. IEEE. doi:10.1109/INFCOMW.2013.6970721.

[90] Syed Hasan, Sergey Gorinsky, Constantine Dovrolis, and Ramesh K. Sitaraman. Trade-offs in optimizing the cache deployments of CDNs. In *IEEE INFOCOM*, pages 460–468, Toronto, Canada, May 2014. IEEE. doi:10.1109/INFOCOM.2014.6847969.

[91] Martin Heusse, Franck Rousseau, Gilles Berger-sabbatel, and Andrzej Duda. Performance Anomaly of 802.11b. In *IEEE INFOCOM*, pages 836–843, San Francisco, CA, USA, March 2003. IEEE.

[92] A K M Mahmudul Hoque, Syed Obaid Amin, Adam Alyyan, Beichuan Zhang, Lixia Zhang, and Lan Wang. NLSR: Named-data Link State Routing Protocol. In *ACM SIGCOMM workshop on Information-centric networking*, page 15, Hong Kong, China, August 2013. ACM. doi:10.1145/2491224.2491231.

[93] Bo Hu, H. Vicky Zhao, and Hai Jiang. Wireless multicast using relays: Incentive mechanism and analysis. *IEEE Transactions on Vehicular Technology*, 62(5):2204–2219, 2013. doi:10.1109/TVT.2012.2236659.

[94] George Iosifidis, Lin Gao, Jianwei Huang, and Leandros Tassiulas. An Iterative Double Auction for Mobile Data Offloading. In *11th International Symposium on Modeling & Optimization in Mobile, Ad Hoc & Wireless Networks (WiOpt)*, pages 154–161, Tsukuba Science City, Japan, May 2013.

[95] George Iosifidis, Lin Gao, Jianwei Huang, and Leandros Tassiulas. Incentive Mechanisms for User-Provided Networks. *IEEE Communications Magazine*, 52(9):20–27, 2014. doi:10.1109/MCOM.2014.6894448.

[96] Alan T. S. Ip, John C. S. Lui, and Jiangchuan Liu. A revenue-rewarding scheme of providing incentive for cooperative proxy caching for media streaming systems. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 4(1):1–32, 2008. doi:10.1145/1324287.1324292.

[97] Van Jacobson. A New Way to Look at Networking, August 2006. Last visited in Aug. 2015. URL: http://named-data.net/a-new-way-to-look-at-networking/.

[98] Van Jacobson, Rebecca L. Braynard, Tim Diebert, Priya Mahadevan, Marc Mosko, Nicholas H. Briggs, Simon Barber, Michael F. Plass, Ignacio Solis, Ersin Uzun, Byoung Joon B J Lee, Myeong Wuk Jang, Dojun Byun, Diana K. Smetters, and James D. Thornton. Custodian-based information sharing. *IEEE Communications Magazine*, 50(7):38–43, 2012. doi:10.1109/MCOM.2012.6231277.

[99] Van Jacobson, Diana K Smetters, Nicholas H Briggs, Michael F Plass, Paul Stewart, James D. Thornton, and Rebecca L Braynard. VoCCN: Voice-over Content-Centric Networks. In *Workshop on Re-architecting the internet - ReArch '09*, page 1, New York, New York, USA, 2009. ACM Press. doi:10.1145/1658978.1658980.

[100] Van Jacobson, Diana K Smetters, Nicholas H Briggs, James D Thornton, Michael F Plass, and Rebecca L Braynard. Networking Named Content. In *ACM CoNEXT*, pages 1–12, Rome, Italy, December 2009. ACM. doi:10.1145/1658939.1658941.

[101] Raj Jain. *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*. Wiley professional computing. Wiley, 1991.

[102] Sugih Jamin, Cheng Jin Cheng Jin, Yixin Jin Yixin Jin, Danny Raz, Yuval Shavitt, and Lixia Zhang. On the placement of Internet instrumentation. In *IEEE INFOCOM*, volume 1, pages 295–304, Tel Aviv, Israel, March 2000. IEEE. doi:10.1109/INFCOM.2000.832199.

[103] Abdallah Jarray and Ahmed Karmouch. Decomposition Approaches for Virtual Network Embedding With One-Shot Node and Link Mapping. *IEEE/ACM Transactions on Networking*, To appear:1–14, 2014.

[104] Juncheng Jia, Qian Zhang, Qin Zhang, and Mingyan Liu. Revenue generation for truthful spectrum auction in dynamic spectrum access. In *ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, page 3, New Orleans, Louisiana, USA, May 2009. ACM. doi:10.1145/1530748.1530751.

[105] Jin Jiang and Claudio Casetti. Socially-aware gateway-based content sharing and backup. In *Proceedings of the 2nd ACM SIGCOMM workshop on Home networks - HomeNets '11*, page 61, 2011. doi:10.1145/2018567.2018582.

[106] Jinjin Liang, Jian Jiang, Haixin Duan, Kang Li, Tao Wan, and Jianping Wu. When HTTPS Meets CDN: A Case of Authentication in Delegated Service. In *IEEE Symposium on Security and Privacy*, pages 67–82, San Jose, California, USA, May 2014. IEEE. doi:10.1109/SP.2014.12.

[107] Gaurav S. Kasbekar and Saswati Sarkar. Spectrum auction framework for access allocation in cognitive radio networks. *IEEE/ACM Transactions on Networking*, 18(6):1841–1854, 2010. doi:10.1109/TNET.2010.2051453.

[108] Kostas Katsalis, Vasilis Sourlas, Thanasis Papapioannou, Thanasis Korakis, and Leandros Tassiulas. Content Placement in Heterogeneous End-to-End Virtual Networks. In *ACM Symposium On Applied Computing (SAC)*, page 7, Salamanca, Spain, April 2015. ACM.

[109] Olivier Klopfenstein. A randomized rounding heuristic to reroute tunnels in MPLS networks. In *5th International Workshop on Design of Reliable Communication Networks*, pages 461–467, Ischia, Naples, Italy, October 2005. doi:10.1109/DRCN.2005.1563909.

[110] Stavros G. Kolliopoulos and Clifford Stein. Approximation Algorithms for Single-Source Unsplittable Flow. *SIAM Journal on Computing*, 31(3):919–946, 2001. doi:10.1137/S0097539799355314.

[111] Jakub Konka, Ivan Andonovic, Craig Michie, and Robert Atkinson. Auction-based network selection in a market-based framework for trading wireless com-

munication services. *IEEE Transactions on Vehicular Technology*, 63(3):1365–1377, 2014. doi:10.1109/TVT.2013.2280344.

[112] Teemu Koponen, Mohit Chawla, Byung-gon Chun, Andrey Ermolinskiy, Kye Hyun Kim, Scott Shenker, and Ion Stoica. A data-oriented (and beyond) network architecture. *ACM SIGCOMM Computer Communication Review*, 37(4):181, 2007. arXiv:1282402, doi:10.1145/1282427.1282402.

[113] P. Krishnan, Danny Raz, and Yuval Shavitt. The cache location problem. *IEEE/ACM Transactions on Networking*, 8(5):568–582, 2000. doi:10.1109/90.879344.

[114] Derek Kulinski and Jeff Burke. NDNVideo : Random-access Live and Pre-recorded Streaming using NDN. Technical Report September, NDN, September 2012.

[115] Jun Kurihara, Ersin Uzun, and Christopher A Wood. An Encryption-Based Access Control Framework for Content-Centric Networking. In *IFIP Networking Conference*, Toulouse, France, May 2015.

[116] I. Landa-Torres, S. Gil-Lopez, J. Del Ser, S. Salcedo-Sanz, D. Manjarres, and J. a. Portilla-Figueras. Efficient citywide planning of open WiFi access networks using novel grouping harmony searchheuristics. *Engineering Applications of Artificial Intelligence*, 26(3):1124–1130, 2013. doi:10.1016/j.engappai.2012.05.020.

[117] Hyun Yong Lee and Akihiro Nakao. User-assisted in-network caching in information-centric networking. *Computer Networks*, 57(16):3142–3153, 2013. doi:10.1016/j.comnet.2013.07.008.

[118] Kyunghan Lee, Joohyun Lee, Yung Yi, Injong Rhee, and Song Chong. Mobile data offloading: How much can wifi deliver? *IEEE/ACM Transactions on Networking*, 21(2):536–550, 2013. doi:10.1109/TNET.2012.2218122.

[119] Barry M. Leiner, Vinton G. Cerf, David D. Clark, Robert E. Kahn, Leonard Kleinrock, Daniel C. Lynch, Jon Postel, Lawrence G. Roberts, and Stephen S. Wolff. The past and future history of the Internet. *Communications of the ACM*, 40(2):102–108, 1997. arXiv:0904.0395, doi:10.1145/253671.253741.

[120] Zhe Li and Gwendal Simon. Time-shifted TV in content centric networks: The case for cooperative in-network caching. In *IEEE International Conference on Communications (ICC)*, pages 1550–3607, Kyoto, Japan, June 2011. doi:10.1109/icc.2011.5963380.

[121] Hongqiang Harry Liu, Ye Wang, Yang Richard Yang, Hao Wang, and Chen Tian. Optimizing cost and performance for content multihoming. *ACM SIGCOMM Computer Communication Review*, 42(4):371, 2012. doi:10.1145/2377677.2377753.

[122] Xi Liu, Florin Dobrian, Henry Milner, Junchen Jiang, Vyas Sekar, Ion Stoica, and Hui Zhang. A case for a coordinated internet video control plane. *ACM SIGCOMM Computer Communication Review*, 42(4):359, 2012. doi:10.1145/2377677.2377752.

[123] Xian Liu. The role of stochastic programming in communication network design. *Computers & Operations Research*, 32(9):2329–2349, 2005. doi:10.1016/j.cor.2004.03.006.

[124] Priya Mahadevan, Ersin Uzun, Spencer Sevilla, and J.J. Garcia-Luna-Aceves. CCN-KRS: A Key Resolution Service for CCN. In *1st International Conference on Information-Centric Networking - (ICN)*, volume 94304, pages 97–106, Paris, France, September 2014. AC. doi:10.1145/2660129.2660154.

[125] Michele Mangili, Fabio Martignon, and Antonio Capone. A comparative study of content-centric and content-distribution networks: Performance and bounds. In *IEEE Global Communications Conference (GLOBECOM)*, pages 1403–1409, Atlanta, GA, USA, December 2013. IEEE. doi:10.1109/GLOCOM.2013.6831270.

[126] Michele Mangili, Fabio Martignon, Antonio Capone, and Federico Malucelli. Content-aware planning models for information-centric networking. In *IEEE Global Communications Conference (GLOBECOM)*, pages 1854–1860, Austin, TX, USA, December 2014. IEEE. doi:10.1109/GLOCOM.2014.7037078.

[127] Ahmed Mansy and Mostafa Ammar. Analysis of adaptive streaming for hybrid CDN/P2P live video systems. In *International Conference on Network Protocols, ICNP*, pages 276–285, Vancouver, BC, Canada, October 2011. IEEE. doi:10.1109/ICNP.2011.6089062.

[128] Silvano Martello and Paolo Toth. *Knapsack Problems: Algorithms and Computer Implementations*. John Wiley & Sons, Inc., 1990.

[129] Michael Meisel, Vasileios Pappas, and Lixia Zhang. Ad hoc networking via Named Data. In *ACM International Workshop on Mobility in the Evolving Internet Architecture (MobiArch)*, pages 3–8, Chicago, IL, USA, September 2010. ACM. doi:10.1145/1859983.1859986.

[130] Michael Meisel, Vasileios Pappas, and Lixia Zhang. Listen first, broadcast later: Topology-agnostic forwarding under high dynamics. In *Annual Conference of International Technology Alliance in Network and Information Science*, pages 1–8, Imperial College, London, August 2010. URL: http://named-data.net/wp-content/uploads/10ITA-LFBL.pdf.

[131] N. Blefari Melazzi, G. Bianchi, A. Caponi, and A. Detti. A general, tractable and accurate model for a cascade of LRU caches. *IEEE Communications Letters*, 18(5):877–880, 2014. arXiv:arXiv:1309.0718v1, doi:10.1109/LCOMM.2014.031414.132727.

[132] Hiroki Mihara. Content Aware Routing : A Content Oriented Tra ffi c Engineering. In *IEEE Global Communications Conference (GLOBECOM)*, pages 1416–1421, Atlanta, GA, USA, December 2013. IEEE.

[133] Satyajayant Misra, Reza Tourani, and Nahid Ebrahimi Majd. Secure Content Delivery in Information-Centric Networks: Design, Implementation, and Analyses. In *ACM SIGCOMM workshop on Information-*

centric networking, pages 73–78, Hong Kong, China, August 2013. ACM. doi:10.1145/2491224.2491228.

[134] Dov Monderer and Lloyd S. Shapley. Potential Games. *Games and Economic Behavior*, 14(1):124–143, 1996. doi:10.1006/game.1996.0044.

[135] Andre Moreira, Josilene Moreira, Djamel Sadok, Arthur Callado, Moises Rodrigues, Marcio Neves, Victor Souza, and Per P. Karlsson. A case for virtualization of Content Delivery Networks. In *Proceedings of the Winter Simulation Conference (WSC)*, pages 3178–3189, Phoenix, AZ, USA, December 2011. IEEE. doi:10.1109/WSC.2011.6148016.

[136] NDN Project Team. NDN Technical Memo: Naming Conventions. Technical report, NDN, July 2014.

[137] Noam Nisan, Tim Roughgarden, Elsa Tardos, and Vijay V. Vazirani. *Algorithmic game theory*. Cambridge University Press, 2007. arXiv:0907.4385, doi:10.1145/1785414.1785439.

[138] B Niven-Jenkins, F Le Faucheur, and N Bitar. Content Distribution Network Interconnection (CDNI) Problem Statement. RFC 6707 (Informational), 2012. URL: http://www.ietf.org/rfc/rfc6707.txt.

[139] Dusit Niyato and Ekram Hossain. Dynamics of Network Selection in Heterogeneous Wireless Networks: An Evolutionary Game Approach. *IEEE Transactions on Vehicular Technology*, 58(4):2008–2017, 2009. doi:10.1109/TVT.2008.2004588.

[140] Erik Nygren, Ramesh K. Sitaraman, and Jennifer Sun. The Akamai network: a platform for high-performance internet applications. *ACM SIGOPS Operating Systems Review*, 44(3):2–19, 2010. doi:10.1145/1842733.1842736.

[141] By George Pallis and Athena Vakali. Content delivery networks. *Commuications of the ACM*, 49(1):101–106, 2006. doi:10.1145/1107458.1107462.

[142] George Pallis. Improving content delivery by exploiting the utility of CDN servers. *Lecture Notes in Computer Science*, 7450 LNCS:88–99, 2012. doi:10.1007/978-3-642-32344-7_8.

[143] Jianli Pan, Subharthi Paul, and Raj Jain. A survey of the research on future internet architectures. *Communications Magazine, IEEE*, (July):26–36, 2011. URL: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5936152.

[144] Tuan-minh Pham, Serge Fdida, and Panayotis Antoniadis. Pricing in Information-Centric Network Interconnection. In *IFIP Networking Conference*, pages 1–9, Brooklyn, New York, USA, May 2013.

[145] Ioannis Psaras, Wei Koong Chai, and George Pavlou. Probabilistic in-network caching for information-centric networks. In *ACM SIGCOMM workshop on Information-centric networking*, page 55, Helsinki, Finland, August 2012. ACM. doi:10.1145/2342488.2342501.

[146] Xiuquan Qiao, Guoshun Nan, Yue Peng, Lei Guo, Jingwen Chen, Yunlei Sun, and Junliang Chen. NDNBrowser: An extended web browser for named data

networking. *Journal of Network and Computer Applications*, 50:134–147, 2014. doi:10.1016/j.jnca.2014.06.009.

[147] Xiuquan Qiao, Guoshun Nan, Wei Tan, Lei Guo, Junliang Chen, Wei Quan, and Yukai Tu. CCNxTomcat: An extended web server for Content-Centric Networking. *Computer Networks*, 75:276–296, 2014. doi:10.1016/j.comnet.2014.10.014.

[148] Lili Qiu, Venkata N. Padmanabhan, and Geoffrey M. Voelker. On the placement of Web server replicas. In *IEEE INFOCOM*, volume 3, pages 1587–1596, Anchorage, Alaska, USA, April 2001. IEEE. doi:10.1109/INFCOM.2001.916655.

[149] Jarno Rajahalme, Mikko Särelä, Pekka Nikander, and Sasu Tarkoma. Incentive-compatible caching and peering in data-oriented networks. In *ACM CoNEXT*, pages 1–6, Madrid, Spain, December 2008. ACM. doi:10.1145/1544012.1544074.

[150] Jacob Ratkiewicz, Filippo Menczer, Santo Fortunato, Alessandro Flammini, and Alessandro Vespignani. Traffic in social media II: Modeling bursty popularity. In *IEEE International Conference on Social Computing (SocialCom)*, pages 393–400, Minneapolis, MN, USA, August 2010. IEEE. doi:10.1109/SocialCom.2010.63.

[151] Jing Ren, Lemin Li, Huan Chen, Sheng Wang, Shizhong Xu, Gang Sun, Jin Wang, and Shucheng Liu. On the deployment of information-centric network: Programmability and virtualization. In *International Conference on Computing, Networking and Communications (ICNC)*, pages 690–694, Garden Grove, CA, USA, February 2015. IEEE. doi:10.1109/ICCNC.2015.7069429.

[152] Elisha J. Rosensweig and Jim Kurose. Breadcrumbs: Efficient, best-effort content location in cache networks. In *IEEE INFOCOM*, pages 2631–2635, Rio de Janeiro, Brazil, April 2009. IEEE. doi:10.1109/INFCOM.2009.5062201.

[153] Elisha J. Rosensweig, Daniel S. Menasche, and Jim Kurose. On the steady-state of cache networks. In *IEEE INFOCOM*, pages 863–871, Turin, Italy, April 2013. IEEE. doi:10.1109/INFCOM.2013.6566874.

[154] Dario Rossi and Giuseppe Rossini. Caching performance of content centric networks under multi-path routing (and more). Technical report, Telecom ParisTech, 2011. URL: http://www.enst.fr/~drossi/paper/rossi11ccn-techrep1.pdf.

[155] Dario Rossi and Giuseppe Rossini. On sizing CCN content stores by exploiting topological information. In *IEEE INFOCOM Workshops*, pages 280–285, Orlando, FL, USA, March 2012. IEEE. doi:10.1109/INFCOMW.2012.6193506.

[156] Giuseppe Rossini and Dario Rossi. Evaluating CCN multi-path interest forwarding strategies. *Computer Communications*, 36(7):771–778, 2013. doi:10.1016/j.comcom.2013.01.008.

[157] Giuseppe Rossini and Dario Rossi. Coupling caching and forwarding. In *1st International Conference on Information-Centric Networking - (ICN)*, number Lcd, pages 127–136, Paris, France, September 2014. ACM Press. doi:10.1145/2660129.2660153.

[158] Lorenzo Saino, Ioannis Psaras, and George Pavlou. Icarus: a Caching Simulator for Information Centric Networking (ICN). In *International ICST Conference on Simulation Tools and Techniques*, pages 66–75, Lisbon, Portugal, March 2014. ACM. doi:10.4108/icst.simutools.2014.254630.

[159] Stefano Salsano, Nicola Blefari-Melazzi, Francesco Lo Presti, Giuseppe Siracusano, and Pier Luigi Ventre. Generalized virtual networking: An enabler for service centric networking and network function virtualization. In *International Telecommunications Network Strategy and Planning Symposium (Networks)*, volume September, pages 1–7, Funchal, Portugal, September 2014. IEEE. arXiv:arXiv:1409.5257, doi:10.1109/NETWKS.2014.6958523.

[160] Sandvine. Global Internet Phenomena Report - 2H 2014. Technical report, Sandvine, 2015.

[161] Arjuna Sathiaseelan, Richard Mortier, Murray Goulden, Christian Greiffenhagen, Milena Radenkovic, Jon Crowcroft, and Derek McAuley. A Feasibility Study of an In-the-Wild Experimental Public Access WiFi Network. In *ACM Symposium on Computing for Development - ACM DEV*, pages 33–42, San Jose, California, USA, December 2014. ACM. doi:10.1145/2674377.2674383.

[162] Shamik Sengupta and Mainak Chatterjee. An economic framework for dynamic spectrum access and service pricing. *IEEE/ACM Transactions on Networking*, 17(4):1200–1213, 2009. doi:10.1109/TNET.2008.2007758.

[163] Ivan Seskar, Kiran Nagaraja, Sam Nelson, and Dipankar Raychaudhuri. MobilityFirst future internet architecture project. In *7th Conference on Asian Internet Engineering (AINTEC)*, pages 1–5, Bangkok, Thailand, October 2011. ACM. doi:10.1145/2089016.2089017.

[164] Wentao Shang, Jeff Thompson, Jeff Burke, and Lixia Zhang. Development and Experimentation with NDN-JS , a JavaScript Library for Named Data Networking. Technical report, NDN, August 2013.

[165] Wentao Shang, Jeff Thompson, Meki Cherkaoui, Jeff Burke, and Lixia Zhang. NDN. JS: a JavaScript client library for Named Data Networking. In *IEEE INFOCOM Workshops*, pages 1–6, Turin, Italy, April 2013. IEEE. URL: http://new.named-data.net/wp-content/uploads/NOMEN13-ndnjs.pdf.

[166] Sarabjot Singh, Harpreet S. Dhillon, and Jeffrey G. Andrews. Offloading in Heterogeneous Networks: Modeling, Analysis, and Design Insights. *IEEE Transactions on Wireless Communications*, 12(5):2484–2497, May 2013. arXiv:1208.1977, doi:10.1109/TWC.2013.040413.121174.

[167] Diana Smetters, Philippe Golle, and Jim Thornton. CCNx Access Control Specifications. Technical report, Palo Alto Research Center (PARC), July 2010.

[168] Diana Smetters and Van Jacobson. Securing network content. Technical report, Palo Alto Research Center (PARC), October 2009. URL: https://www.parc.com/content/attachments/TR-2009-01.pdf.

[169] Won So, Ashok Narayanan, David Oran, and Mark Stapp. Named data networking on a router: Forwarding at 20Gbps and Beyond. In *ACM SIGCOMM*, pages 495–496, Helsinki, Finland, August 2013. ACM. doi:10.1145/2486001.2491699.

[170] Vasilis Sourlas, Lazaros Gkatzikis, Paris Flegkas, and Leandros Tassiulas. Distributed Cache Management in Information-Centric Networks. *IEEE Transactions on Network and Service Management*, 10(3):286–299, September 2013. doi:10.1109/TNSM.2013.052113.120382.

[171] Statista. Cumulative number of apps downloaded from the Apple App Store from July 2008 to October 2014. Last visited in Aug. 2015. URL: http://www.statista.com/statistics/263794/number-of-downloads-from-the-apple-app-store/.

[172] Yi Sun, Seyed Kaveh Fayaz, Yang Guo, Vyas Sekar, Yun Jin, Mohamed Ali Kaafar, and Steve Uhlig. Trace-Driven Analysis of ICN Caching Algorithms on Video-on-Demand Workloads. In *ACM International on Conference on emerging Networking Experiments and Technologies - CoNEXT*, pages 363–376, Sydney, Australia, December 2014. ACM Press. doi:10.1145/2674005.2675003.

[173] Ruben Torres, Alessandro Finamore, Jin Ryong Kim, Marco Mellia, Maurizio M. Munafò, and Sanjay Rao. Dissecting video server selection strategies in the YouTube CDN. In *IEEE International Conference on Distributed Computing Systems (ICDCS)*, pages 248–257, Minneapolis, Minnesota, USA, June 2011. IEEE. doi:10.1109/ICDCS.2011.43.

[174] Michele Tortelli, Dario Rossi, Gennaro Boggia, and Luigi Alfredo Grieco. CCN simulators. In *1st International Conference on Information-Centric Networking - (ICN)*, pages 197–198, Paris, France, September 2014. ACM. doi:10.1145/2660129.2660133.

[175] By Gareth Tyson, Nishanth Sastry, and Ruben Cuevas. A Survey of Mobility in Centric Networks. *Communications of the ACM*, 56(12):90—-98, 2013. doi:10.1145/2500501.

[176] Pascal Van Hentenryck. *The OPL optimization programming language*. MIT Press, Cambridge, MA, USA, 1999.

[177] RM Van Slyke and Roger J.-B. Wets. L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal on Applied Mathematics*, 17(4):638–663, 1969. URL: http://www.jstor.org/stable/2099310.

[178] Sarut Vanichpun and Armand M. Makowski. The output of a cache under the independent reference model. *ACM SIGMETRICS Performance Evaluation Review*, 32(1):295, 2004. doi:10.1145/1012888.1005722.

[179] Matteo Varvello, Diego Perino, and Jairo Esteban. Caesar: a Content Router for High Speed Forwarding. In *ACM SIGCOMM workshop on Information-centric networking*, pages 73–78, Helsinki, Finland, August 2012. ACM. doi:10.1145/2342488.2342505.

[180] Jason Min Wang, Jun Zhang, and Brahim Bensaou. Content multi-homing: An alternative approach. In *IEEE International Conference on Communications (ICC)*, pages 3118–3123, Sydney, Australia, June 2014. IEEE. doi:10.1109/ICC.2014.6883800.

[181] Lan Wang, A K M Mahmudul Hoque, Cheng Yiy, Adam Alyyan, and Beichuan Zhangy. OSPFN: An OSPF Based Routing Protocol for Named Data Networking. Technical report, NDN, July 2012. URL: http://named-data.net/publications/techreports/trospfn/.

[182] Liang Wang, Suzan Bayhan, and Jussi Kangasharju. Effects of Cooperation Policy and Network Topology on Performance of In-Network Caching. *IEEE Communications Letters*, 18(4):680–683, April 2014. arXiv:1312.0133, doi:10.1109/LCOMM.2014.013114.132647.

[183] Lijing Wang, Ilya Moiseenko, and Lixia Zhang. NDNlive and NDNtube : Live and Prerecorded Video Streaming over NDN. Technical report, NDN, April 2015.

[184] Sen Wang, Jun Bi, Jianping Wu, Xu Yang, and Lingyuan Fan. On adapting HTTP protocol to content centric networking. In *7th International Conference on Future Internet Technologies - CFI '12*, page 1, Seoul, Korea, September 2012. ACM. doi:10.1145/2377310.2377312.

[185] Yonggong Wang, Zhenyu Li, Gareth Tyson, Steve Uhlig, and Gaogang Xie. Optimal cache allocation for content-centric networking. In *International Conference on Network Protocols, ICNP*, pages 1–10, Göttingen, Germany, October 2013. IEEE. doi:10.1109/ICNP.2013.6733577.

[186] Christopher a Wood and Ersin Uzun. Flexible End-to-End Content Security in CCN. In *Consumer Communications and Networking Conference (CCNC)*, pages 872–879, Las Vegas, NE, USA, January 2014. IEEE. doi:10.1109/CCNC.2014.6940528.

[187] Mengjun Xie, Indra Widjaja, and Haining Wang. Enhancing cache robustness for content-centric networking. In *IEEE INFOCOM*, pages 2426–2434, Orlando, FL, USA, March 2012. IEEE. doi:10.1109/INFCOM.2012.6195632.

[188] Aifang Xu, Xiaodong Tan, and Ye Tian. Design and Evaluation of a Utility-based Caching Mechanism for Information-centric Networks. In *IEEE International Conference on Communications (ICC)*, pages 7138–7143, London, UK, June 2015. IEEE.

[189] Dongyan Xu, Sunil Suresh Kulkarni, Catherine Rosenberg, and Heung Keung Chai. Analysis of a CDN-P2P hybrid architecture for cost-effective streaming media distribution. *Multimedia Systems*, 11(4):383–399, 2006. doi:10.1007/s00530-006-0015-3.

[190] Hongfeng Xu, Zhen Chen, Rui Chen, and Junwei Cao. Live streaming with content centric networking. In *International Conference on Networking and Distributed Computing, ICNDC*, pages 1–5, Hangzhou, China, October 2012. IEEE. doi:10.1109/ICNDC.2012.9.

[191] George Xylomenos, Christopher N Ververidis, Vasilios A Siris, Nikos Fotiou, Christos Tsilopoulos, Xenofon Vasilakos, Konstantinos V Katsaros, and George C Polyzos. A Survey of information-centric networking research. *IEEE Communications Surveys and Tutorials*, 16(2):1024–1049, 2014. doi:10.1109/SURV.2013.070813.00063.

[192] Cheng Yi, Jerald Abraham, Alexander Afanasyev, Lan Wang, Beichuan Zhang, and Lixia Zhang. On the Role of Routing in Named Data Networking. Technical report, NDN, December 2013. doi:10.1145/2660129.2660140.

[193] Cheng Yi, Alexander Afanasyev, Lan Wang, Beichuan Zhang, and Lixia Zhang. Adaptive forwarding in named data networking. *ACM SIGCOMM Computer Communication Review*, 42(3):62, 2012. doi:10.1145/2317307.2317319.

[194] Guoqiang Zhang, Yang Li, and Tao Lin. Caching in information centric networking: A survey. *Computer Networks*, 57(16):3128–3141, 2013. doi:10.1016/j.comnet.2013.07.007.

[195] Xinwen Zhang, Katharine Chang, Huijun Xiong, Yonggang Wen, Guangyu Shi, and Guoqiang Wang. Towards name-based trust and security for content-centric network. In *International Conference on Network Protocols, ICNP*, pages 1–6, Vancouver, BC, Canada, October 2011. IEEE. doi:10.1109/ICNP.2011.6089053.

[196] Yu Zhang, Hongli Zhang, and Lixia Zhang. Kite: A Mobility Support Scheme for NDN. In *1st International Conference on Information-Centric Networking - (ICN)*, pages 179–180, Paris, France, September 2014. ACM. doi:10.1145/2660129.2660159.

[197] Xia Zhou and Heather Zheng. TRUST: A general framework for truthful double spectrum auctions. In *IEEE INFOCOM*, pages 999–1007, Rio de Janeiro, Brazil, April 2009. IEEE. doi:10.1109/INFCOM.2009.5062011.

[198] Kun Zhu, Dusit Niyato, Ping Wang, and Zhu Han. Dynamic spectrum leasing and service selection in spectrum secondary market of cognitive radio networks. *IEEE Transactions on Wireless Communications*, 11(3):1136–1145, 2012. doi:10.1109/TWC.2012.010312.110732.

[199] Zhenkai Zhu and Alexander Afanasyev. Let's ChronoSync: Decentralized dataset state synchronization in Named Data Networking. In *IEEE International Conference on Network Protocols (ICNP)*, pages 1–10, Göttingen, Germany, October 2013. IEEE. doi:10.1109/ICNP.2013.6733578.

[200] Zhenkai Zhu, Alexander Afanasyev, and Lixia Zhang. A new perspective on mobility support. Technical report, NDN, July 2013. URL: http://new.named-data.net/wp-content/uploads/TRmobility.pdf.

[201] Zhenkai Zhu, Jeffery Burke, Lixia Zhang, Paolo Gasti, Yanbin Lu, and Van Jacobson. A New Approach to Securing Audio Conference Tools. In *Asian Internet Engineering Conference (AINTEC)*, pages 120–123, Bangkok, Thailand, November 2011. ACM. doi:10.1145/2089016.2089036.

[202] Zhenkai Zhu, Sen Wang, Xu Yang, Van Jacobson, and Lixia Zhang. ACT: Audio Conference Tool Over Named Data Networking. In *ACM SIGCOMM workshop on Information-centric networking*, volume 11, page 68, New York, New York, USA, 2011. ACM Press. doi:10.1145/2018584.2018601.